Multi-Cue People Detection from Video

Dissertation

approved by the Department of Computer Science,
TECHNISCHE UNIVERSITÄT DARMSTADT

in fulfillment of the requirements for the degree of
Doktor-Ingenieur (Dr.-Ing.)
by
Dipl.-Phys. Stefan Walk
from Diez a.d. Lahn, Germany

Referee:	Prof. Stefan Roth, Ph.D.
Co-referee:	Prof. Dr. Bernt Schiele
Co-referee:	Prof. Dr. Konrad Schindler

Date of submission: 6th of August, 2012
Date of oral examination: 26th of September, 2012

Darmstadt 2013
D17

ABSTRACT

This thesis aims to advance the state of the art in pedestrian detection. Since there are many applications for pedestrian detection, for example automotive safety or aiding robot-human interaction in robotics, there is a strong desire for improvement. In this thesis, the benefits of combining multiple features that gather information from different cues (for example image color, motion and depth) are studied. Training techniques and evaluation procedures are also investigated, improving performance and the reliability of results, especially when different methods are compared.

While motion features were previously used, they either were conceptually restricted to a setting with a fixed camera (e.g. surveillance) [100] or were not resulting in an improvement for the full-image detection task [10, 12]. In this thesis, the necessary modifications to the approach of Dalal *et al.* [12] (which is based on optical flow) to make it work in the full-image detection setting are presented. In addition to this, substantial improvements using motion features are shown even when the camera is moving significantly, which has not been tested before. A variant of the motion feature that performs equally well with a significantly lower feature dimension is also introduced.

Another cue that is used in the present work is color information. Usually, when incorporating color information into computer vision algorithms, one has to deal with the color constancy problem. In this thesis, a new feature called color self-similarity (CSS) is introduced. It encodes long-range (between positions within the detector window) similarities of color distributions. By only comparing colors inside the detector window, the color constancy problem can be circumvented – effects of lighting and camera properties are less likely to vary significantly within the detector window than they are over the whole dataset. Additionally, it is shown that even raw color information can be useful if the training set covers enough variability.

Depth is also a useful cue. An existing stereo feature – stereo-based HOG by Rohrbach *et al.* [75] – is adopted and a new feature that exploits a useful relation between stereo disparity and the height of an object in an image is introduced. This feature is computationally cheap and able to encode local scene information, like object height and the presence of a ground plane, in a completely data-driven way (all parameters are learned during training). It helps both by reducing false positives (eliminating those that have the wrong size) and false negatives (those that were missed because the detector estimated the size wrongly).

For the classifier part of the pipeline, it is shown that AdaBoost with decision stumps is not able to handle the multi-cue, multi-view detection setting that we are examining well. A recently proposed boosting classifier, MPLBoost, turned out to be superior, resulting in classification performance that is comparable to support vector machines. It is also demonstrated that error rates can be reduced by using support

vector machines and boosting classifiers in combination. Another contribution of this thesis is a procedure to combine training datasets with different sets of cues during training, e.g. a monochrome dataset with a colored dataset, or a dataset with no motion information with a dataset from video. This greatly increases the amount of available training data when multiple cues are used.

A collection of pitfalls during evaluation is also highlighted. It is demonstrated that the PASCAL overlap criterion encourages overestimating the bounding box size. Care also has to be taken when evaluating on subsets of annotations, e.g. only on occluded pedestrians or pedestrians of certain sizes. When trying to determine the strengths of different approaches, naive approaches can easily lead to wrong conclusions. In this thesis, better methods to compare different approaches are proposed.

An application of the detector in a 3D scene reasoning framework is also presented. Multiple detectors trained on partial (e.g. only upper body) views are combined. 3D reasoning is used to infer which parts of the pedestrian should be visible and the framework uses this information to determine the strengths of the contributions of the partial detectors. This allows the detection system to find pedestrians even when they are occluded for extended periods of time.

ZUSAMMENFASSUNG

Die automatische Detektion von Fußgängern ist ein Forschungsgebiet, das viele Anwendungen hat. Fußgänger-Detektions-Algorithmen liefern zum Beispiel Fahrassistenzsystemen die nötigen Informationen um zu verhindern, dass Fußgänger von einem Auto überfahren werden, und können in der Robotik verwendet werden um die Roboter-Mensch-Interaktion zu verbessern. Ziel dieser Arbeit ist, den Stand der Technik in der Fußgängerdetektion zu verbessern. Dazu wird untersucht, inwieweit die Kombination von mehreren Informationsquellen (z.B. Farbe, Bewegung im Bild und Entfernung) hilfreich ist und wie diese Kombination am besten durchgeführt werden kann. Zusätzlich dazu werden Prozeduren zum Training und zur Evaluierung von Algorithmen untersucht, um die Genauigkeit der Erkennung und die Verlässlichkeit der Ergebnisse, insbesondere wenn mehrere Ansätze verglichen werden, zu erhöhen.

Es gab bereits Ansätze, Bewegungsinformation für die Fußgängererkennung zu nutzen. Diese waren jedoch konzeptbedingt lediglich für Situationen geeignet, in denen man eine feste Kamera hat (wie z.B. in Überwachungsszenarien) [100] oder erreichten keine Verbesserung, wenn sie zur Fußgängerdetektion in ganzen Bildern eingesetzt wurden [10, 12]. In dieser Arbeit werden die nötigen Veränderungen zum Ansatz von Dalal *et al.* [12] (der auf optischem Fluß basiert) dargelegt, um ihn zur Fußgängerdetektion in ganzen Bildern verwenden zu können. Zusätzlich dazu wird gezeigt, dass die Einbindung von Bewegungsinformation in den Detektor zu deutlichen Verbesserungen führt, auch wenn die Kamera sich stark bewegt (wie es in einem Auto der Fall ist).

Farbe ist ebenfalls eine nützliche Informationsquelle, die in dieser Arbeit genutzt wird. Bei der Benutzung von Farbinformation stößt man typischerweise auf das Problem, dass die Farbe, die von der Kamera wahrgenommen wird, außer von der Objektfarbe (die man nutzen möchte) noch von Kameraeigenschaften und vom Licht beeinflusst wird. Das menschliche Gehirn versucht Lichteinflüsse zu ignorieren (Farbkonstanz), und obwohl es Ansätze gibt eine ähnliche Funktionalität für Computer zu ermöglichen ist eine gleichwertige Lösung dieses Problems noch nicht bekannt. In dieser Arbeit wird ein Merkmal namens „Color Self-Similarity“ vorgestellt, das Ähnlichkeiten von Farbverteilungen innerhalb des Detektorfensters kodiert. Dadurch, dass Farben lediglich innerhalb des Detektorfensters verglichen werden (und nicht zwischen verschiedenen Bildern von Fußgängern) kann das Problem, dass Farben von Licht- und Kameraeinflüssen abhängen, umgangen werden (da diese sich innerhalb eines Bildausschnittes das einen Fußgänger enthält in der Regel nicht signifikant ändern). Es wird aber auch gezeigt, dass selbst „rohe“ Farbinformation helfen kann, wenn der Trainingsdatensatz genug Variabilität aufweist.

Eine weitere hilfreiche Informationsquelle ist die Entfernung. Ein neues Merkmal,

das eine feste Beziehung zwischen der Größe eines Objektes im Bild und der Stereo-Disparität ausnutzt, wird in dieser Arbeit vorgestellt. Dieses Merkmal ist schnell zu berechnen und kann lokale Information über den Fußgänger (z.B. seine Größe) und seine Umgebung (z.B. dass Fußgänger üblicherweise auf dem Boden stehen) kodieren. Diese Eigenschaften werden vollständig aus den Trainingsdaten gelernt, so dass das Merkmal ohne Modifikation auch für andere Objektklassen benutzt werden könnte.

In Bezug auf die Klassifizierungsalgorithmen, die für die Fußgängerdetektion genutzt werden, wird gezeigt, dass das populäre AdaBoost mit Entscheidungsbäumen der Tiefe 1 nicht für das Szenario geeignet ist, das wir betrachten (Fußgängerdetektion aus vielen verschiedenen Blickwinkeln mit der Verwendung von mehreren Informationsquellen). Ein anderer Klassifizierungsalgorithmus, MPLBoost, führte zu deutlich verbesserten Erkennungsraten. Es wird gezeigt, dass Support Vector Machines und MPLBoost kombiniert werden können, um Fehler zu reduzieren. Ein weiterer Beitrag dieser Arbeit ist eine Prozedur, um Trainingsdatensätze zu kombinieren, die verschiedene Informationsquellen enthalten (wie z.B. ein farbiger Datensatz und ein Datensatz mit Graustufen oder ein Datensatz aus Videos mit einem Datensatz aus Einzelbildern ohne Bewegungsinformation). Dies führt zu einer deutlich höheren Gesamtdatenmenge, die für das Training genutzt werden kann.

Außerdem werden in dieser Arbeit eine Reihe von Problemen aufgezeigt, die bei der Evaluierung von Algorithmen auftauchen können. Zum Beispiel wird gezeigt, dass das PASCAL-Kriterium (welches angibt, wann eine Detektion einer Annotation zugeordnet werden darf) Methoden bevorzugt, die die Größe des Objekts bzw. des Fußgängers überschätzen. Vorsicht ist auch geboten, wenn es darum geht nur auf einem Teil der Annotationen zu evaluieren (z.B. nur auf teilweise verdeckten Fußgängern). Wenn verschiedene Algorithmen verglichen werden, können naive Arten des Vergleichens leicht zu falschen Schlüssen führen. In dieser Arbeit werden bessere Methoden vorgestellt.

Des Weiteren wird die Verwendung des Detektors in einem 3D-Szenenmodell gezeigt. Mehrere Detektoren, die auf Teile des Fußgängers (z.B. nur den Oberkörper) trainiert worden sind, werden kombiniert. Durch die Benutzung des Szenenmodells können Schlüsse gezogen werden, welche Teile der Fußgänger gerade sichtbar sein sollten und die Gewichtung der einzelnen Detektoren kann angepasst werden. Dadurch kann das System Fußgänger erkennen, auch wenn sie über längere Zeiträume teilweise verdeckt sind, was eine Schwäche von vielen Algorithmen zur Fußgängerdetektion behebt.

ACKNOWLEDGMENTS

I am grateful to everyone who provided me with advice and support during my time as a PhD student.

First of all, I would like to thank my supervisor, Prof. Bernt Schiele, for many ideas and excellent feedback from the start to the end of my PhD thesis. I would also like to thank Prof. Konrad Schindler for co-supervising my work and for many good suggestions. To Prof. Stefan Roth I want to express my gratitude for his suggestions and for agreeing to be the main examiner of my thesis.

I am grateful to the members of the MIS, IU, ESS and GRIS groups at TU Darmstadt and the D2 department at MPI Informatics, Saarbrücken, for lots of fruitful discussions in the offices, at breaks and on retreats. I especially would like to thank Christian Wojek for his help and discussions, particularly at the start of my PhD thesis. Ursula Paeckel was very helpful in administrative and other matters, making life as a PhD student considerably easier.

I am also grateful to Toyota Motor Europe both for funding my research and for providing realistic scenarios in which I could test my approaches.

I also would like to thank all of my friends for the great time I had in Darmstadt, especially Markus and Milena Wagner for being great flatmates and enduring me during stressful times, and Stefanie Sammet for providing a lot of help during the final parts of my PhD thesis (and for being a great friend overall).

Finally, I am very grateful to my parents and family for their continuing support and encouragement, even in times that were troubling for them.

CONTENTS

1 INTRODUCTION

Contents

Computers have become ubiquitous helpers for humans. They provide assistance for many tasks we have to do, be it finding the shortest route to our destination for a car trip, finding a document of which we only have a coarse idea of the contents, synchronizing our calendar with other people's calendars, or finding the cheapest seller for a product we want to buy. Usually, they perform tasks like this at least as good as a person that is not specifically trained for this task, and orders of magnitude faster in addition to that. However, there are some areas that even untrained humans still perform far better than computers. Speech recognition, for example, is now beginning to see widespread usage in smartphones and other devices, however it is still far behind the performance of a human.

The analysis of visual input is another task we humans solve easily and subconsciously, while being exceptionally hard for computers. Figure 1.1 is a collection of images of chairs, taken from an image search engine. While for each of those instances the purpose of the object in the image is clear to us, it is hard for a computer to learn to visually recognize this class of objects because of the large variability in appearance, even for those samples that have almost no background clutter.

The goal of this thesis is to advance the state of the art in the detection of pedestrians, which are an especially challenging "object" class, for reasons that will be laid out later in this chapter. However, most of the results are – more or less directly – applicable to other object classes as well. The remainder of this chapter is structured as follows: In section 1.1 the field of visual object recognition, a sub-area of computer vision, is introduced. Section 1.2 explains the pedestrian detection task and highlights its challenges. An overview of our approach to solve this task is then shown in section 1.3. Finally, section 1.4 provides an outline of the following chapters.

Figure 1.1: The first results of a google image search for "chair". There is a huge visual variability, however for humans it is clear – in most cases intuitively, and in every case after closer inspection – that each of those objects is a chair.

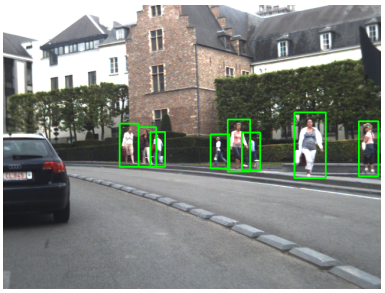
1.1 VISUAL OBJECT RECOGNITION

The field of tasks related to analyzing and understanding an image is called *computer vision*. By "understanding an image" we mean inferring information about the world that produced the image. In this sense, some computer vision tasks can be seen as the inverse of computer graphics – while computer graphics seeks to produce an image given a model of the world, the task of obtaining a model of the world given one or more images belongs to computer vision. However, there are also low-level vision tasks that do not build an explicit model of the world, like determining "this group of pixels was produced by a patch of grass" without seeking information about where in the world this patch of grass is.

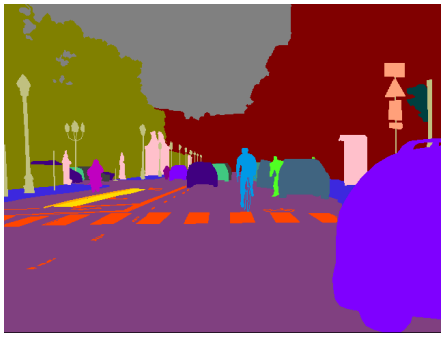
There are many applications for computer vision systems, some of them already in use today. Some examples are the face detection capability of many consumer cameras and the pose and gesture recognition of the Kinect game console. However, most of the methods in use today are not very robust, preventing their use in critical systems. There are applications of computer vision that are currently not feasible, such as a fully autonomous car.

Often, computer vision tasks can be grouped into a set of tasks that have been defined by the computer vision community. Examples of those tasks are:

Image Classification Image classification is the task of generating a high-level description of the image, e.g. for the right-hand image[1] – the result could be "This is an image of a cat." One obvious application for this task would be database queries (image retrieval), e.g. telling a program "Find all images in my personal photo album that contain a cat." Image classification usually deals with *classes* of objects and scenes, the task "Find all images containing **my** cat" is usually termed *recognition* instead of *classification* (because we are searching for a specific instance of the object class, instead of all instances).



Object Detection Object detection is the task of localizing all instances of an object class in an image. This means determining the position and extents of all objects of the given class, usually providing the bounding boxes of the objects as a result, as presented in the image to the right for the example of the object class "pedestrians", which is the main topic of this thesis. Detection obviously provides more detailed results than classification – "There is a pedestrian directly in front of the car" would be a good reason for an autonomous car to stop, while "There is a pedestrian somewhere in view of the camera" is a less helpful piece of information for this application. As with image classification, there is also the special case where we are looking for the position of a specific object instance, discriminating it from all the other instances of this class.



Segmentation Segmentation is the task of assigning each pixel to the object class that "produced" that pixel. A human-produced segmentation for an image is shown to the right. Pedestrians, cars, bicyclists, motor scooters, street markings, buildings and other classes are marked with different colors. Segmentation provides an even more detailed analysis, aiming to explain the whole image. The output of segmentation algorithms can be useful e.g. to reason about occlusion, helping object detectors, or to aid computer graphics tasks such as taking one object and putting it into another scene.

Other, more high-level tasks in computer vision are pose/viewpoint estimation (e.g. estimating the configuration of body parts of a human in an image, which can be used for gesture or action recognition) or 3D scene analysis, where we go beyond analyzing the image and aim to reason about the world that produced the image.

[1] The image is taken from the PASCAL visual objects classes challenge [31].

1.2 PEDESTRIAN DETECTION

Pedestrian detection is one instance of the object detection task. In pedestrian detection, we seek to find every person in the image that is standing, walking or running – sitting persons or people in uncommon poses, as seen for example during sports activities, are excluded. One application, and the main one we had in mind while researching this topic, is pedestrian safety. Every year, over 30 000 pedestrians are injured or killed in traffic accidents in Germany alone [88]. If cars were able to recognize situations where pedestrians are in danger, they could help to prevent accidents by warning the driver or even by braking and evading on their own. In this case, persons that are running, walking or standing are the most likely persons to cross the trajectory of the car – a sitting person is unlikely to change its position in the next couple of seconds. In industrial applications robots or automatic doors could stop if there is a pedestrian approaching. However, there are some challenges that have to be overcome if one wants to perform automatic pedestrian detection. Some of those challenges are common to most object detection tasks, and some of them are specific to pedestrian detection:

Viewpoint variation Pedestrians, like most object classes, look very different if seen from multiple viewpoints. In a surveillance scenario, where the camera is usually mounted in a high place and looking down at the scene, pedestrians look very different from pedestrians seen from a camera mounted on a car. Pedestrians also look differently when viewed from the side instead from the front or back. Depending on the context, different viewpoints may be more or less common. For example, in a photo album, persons are usually looking into the camera and are seen from the front.

Pose variation There is a huge variability in poses that humans can adopt. As mentioned, we are restricting ourself to the "pedestrian" setting for this thesis, where humans are standing or walking upright, as opposed to sitting or lying down, or other non-pedestrian poses that are seen e.g. in sports scenes. Even then, there is a lot of pose variation, especially for side views where the legs are moving significantly during the walking cycle.

Variation between individual humans The former points generate variability for a single human in a specific setting. When we look at different humans, there are many more reasons for high variability:

- Body properties like height, weight, age, or muscularity significantly change the appearance of pedestrians. In fact, kids are often a problematic case for pedestrian detectors because their body proportions differ from those of adults.
- The type of clothing also influences the appearance a lot. Warm clothing in cold regions or in winter tends to hide human features, as do long dresses or skirts, which hide the discriminative shape that two legs provide.

- Clothing is also usually textured – a Hawaiian shirt looks very different from a business shirt. There are almost no completely unrealistic textures for human clothing. There are some very common types of textures, like blue jeans or business suits, but there is also lots of very colorful clothing with custom imagery.

Other factors, like different skin colors, also play a role. While this property (huge intra-class variability) makes it easier to recognize a specific pedestrian (and e.g. track him or her across a video sequence), it makes the general task of detecting pedestrians harder.

Varying scenes The scene, or "background" for the pedestrians is also highly varying. Pedestrians can be found in all kinds of places, e.g. in indoor scenes like supermarkets, in the nature on mountains or beaches and along streets in cities. This affects the difficulty of the task, as it is easier to tell pedestrians apart from the background scenery on a beach than it is to tell them apart from the cluttered background in an inner city. It is also relevant because many detectors incorporate some form of context (the surrounding of the image region in question) to decide if there is a pedestrian present or not. For those detectors, a high variability in background means high intra-class variability.

Different lighting/weather conditions Another challenge in detecting pedestrians are changing lighting conditions, for outdoor scenes often imposed by the weather. Adverse weather conditions include snow, rain, or – not quite intuitively – sunny weather. Sunny weather leads to pronounced shadows, which (in combination with the limited dynamic range of cameras, especially digital ones) lead to overexposed image regions lit by direct sunlight or underexposure in the image regions covered by shadow. A camera looking at the general direction of the sun can also lead to lens flares. For this reason, overcast weather is the most "friendly" condition for pedestrian detection, because it leads to a uniform lighting with few shadows while providing enough light for digital cameras using fast shutter speeds.

Occlusion/Truncation Often, pedestrians are not completely visible. They may be truncated by the image border or some other object may be in front of the pedestrian. In the setting of detecting pedestrians in street scenes, most occlusions are caused by other pedestrians (e.g. if pedestrians are walking in a group) or by cars. The occluding objects may also belong to the pedestrian in question – he or she might be carrying a suitcase or an umbrella.

Resolution Pedestrians occur at many scales in images. In a street scene, pedestrians can appear so big that they are truncated at the top and the bottom of the image if they are very close to the car (e.g. when they are on a crosswalk in front of the car) and arbitrarily small when they are far away. Both extremes pose unique challenges: truncated pedestrians often lack distinctive features like face or legs, and far-away pedestrians are only a few pixels in size, making

it hard to discriminate them from other structures with roughly pedestrian-like shapes.

1.3 OVERVIEW

This section introduces the pedestrian detection method[2] that is employed in this thesis. In order to make the pedestrian detection task more manageable, it is common to replace the question "Where in this image are pedestrians?" with the more easily handled question "Is there a pedestrian *at this position (and scale)* in the image?", asking this question at many points in the image.

In our case, we adopt the *sliding window* method, testing each position of an evenly spaced, overlapping grid on the image. An image region (the detector window) around the position is extracted[3] and tested for the presence of a pedestrian. The window then moves (slides) to the next position, where the process is repeated. After each point has been processed, the image is scaled down by a certain factor (the scale step) and the process is repeated until the image is smaller than the detector window.

As the grid is usually significantly overlapping, multiple positive responses can be expected for each pedestrian (as he or she is covered in slightly different positions in the detector window and in slightly different sizes). Thus, the multiple detections need to be merged. This is done by clustering the detections, leaving only the most confident detection of a cluster.

By downscaling the image instead of upscaling the detector window, the algorithm we use to decide if the window contains a pedestrian or not only has to deal with image regions of a fixed size, simplifying the task. As a consequence of that, the minimal object size one can detect is the size of the detector window. In order to detect smaller objects, one can either reduce the size of the detector window or enlarge the image before starting the sliding window search.

The decision if a pedestrian is present or not is of binary (two-class) classification task: We want to tell the class of pedestrians (generally "foreground" or "positive class") apart from the class of non-pedestrians ("background" or "negative class"). Usually, machine learning is employed in some form for this binary classification task. Conceptually, this decision process is dividable into two steps:

1. A *feature transform* is applied to the image region. The transform maps image regions into a feature space X, which in our case (but not necessarily) is a subset of $\mathbb{R}^N$, where N is called *feature dimension*, and the elements of X are called *feature vectors*.

[2] Although it is a general object detection method, pedestrians are used as an example object class throughout this section.

[3] If the region one wants to test extends beyond the image borders, one usually extrapolates the missing pixels from the pixels at the border of the image.

2. A classifier $f : X \times A \to \mathbb{R}$ is used to map the feature vector to a confidence rating. A higher $f(\boldsymbol{x}; \boldsymbol{\alpha})$ means that the classifier is more confident that the feature vector $\boldsymbol{x} \in X$ belongs to the positive class[4]. $\boldsymbol{\alpha} \in A$ is a set of parameter values for the classifier.

The point of the feature transform is to eliminate meaningless variability (such as changes in lighting) while preserving information that is needed to discriminate between classes in order to make the classifiers' task easier. Features often focus on capturing one source of information – like shape, texture, movement or depth. In order to utilize multiple cues, one then combines those features. In order to achieve robustness against irrelevant variability, steps like spatial smoothing, normalization and quantization are often integral parts of feature mapping schemes.

Ideally, the property of being a "good feature" would be independent from the task at hand. This way, one could keep the features fixed across tasks and only learn a different classification function. However, in practice a feature that is good for discriminating between birds and fishes will probably not be a good feature for discriminating between trout and codfish (because the properties that are important to tell them apart would fall in the "meaningless" category for the first task). A good feature for all tasks would (among other things) have to capture this hierarchical structure of object classes, which is probably a harder task than object detection itself.

As previously mentioned, machine learning algorithms are the method of choice for the classification step. That means that the set of parameter values $\boldsymbol{\alpha}$ is not given *a priori* but learned during a *training phase*. During training, a set of feature vectors $\boldsymbol{x_i}$ (the training examples) with corresponding labels $y_i \in \{+1, -1\}$ (signaling whether the sample belongs to the positive or the negative class) is used. The parameters are then chosen by minimizing a penalty or *loss function* over the training set:

$$\boldsymbol{\alpha} = \arg\min_{\boldsymbol{\alpha}'} L\left(f(\,\cdot\,; \boldsymbol{\alpha}'), \{(\boldsymbol{x_i}, y_i)\}\right) \tag{1.1}$$

The simplest example of a loss function (simplest in structure, not simplest to optimize) is the error rate. By thresholding the classifier outputs (e.g. at $\theta = 0.5$ for classifiers that output probability estimates), we obtain the estimated labels:

$$\hat{y}_i = \begin{cases} +1 & \text{if} \quad f(\boldsymbol{x_i}; \boldsymbol{\alpha}) \geq \theta \\ -1 & \text{if} \quad f(\boldsymbol{x_i}; \boldsymbol{\alpha}) < \theta \end{cases} \tag{1.2}$$

The error rate is then simply the fraction of samples where $y_i \neq \hat{y}_i$. However, the error rate is not convex as a function of the output of f (it is not even continuous), so it is hard to optimize. Practical loss functions usually combine a convex upper bound of the error rate with a regularizing term. Regularization is applied to improve the

[4]This monotony is the only requirement that we impose in general. The scores don't need to be calibrated in some way. Some classifiers output probability estimates ranging from 0 to 1, other classifiers' outputs could cover the entire range of real numbers.

generalization of the classifier, aiming to ensure that low classification error rates on the training set translate to good performance on samples that are not in the training set.

The usage of the classifier to determine the label of a given sample is referred to as *testing*[5]. In order to see how good a combination of feature and classifier is, we train it on a training set and then evaluate it on a distinct test set. By doing this instead of just measuring the performance on the training set we ensure that the feature/classifier combination captured characteristics of the object class that generalize beyond the training set, instead of just the characteristics of the samples that were available during training. The classifiers' main assumption is that the feature vectors that it sees during training and testing are independently sampled from the same distribution, and this assumption usually does not hold for real training and test sets, e.g. because of different cameras, weather or viewpoints, or because of regional biases or different locations, e.g. pedestrian zone vs. street. Because of this, in such scenarios it is vital for a good feature to discard variability caused by dataset bias in order to get good performance on the test set.

The aspects of pedestrian detection that we will focus on most in this thesis are feature computation, the choice and training of classifiers and the evaluation of the resulting detector.

1.4 OUTLINE

Chapter 2 – Related Work
This chapter provides an overview of the related work in the field of pedestrian detection, and other publications that influenced this thesis.

Chapter 3 – Multi-Cue Onboard Pedestrian Detection
In this chapter, we investigate the use of motion features in a pedestrian detection system with a moving camera, and demonstrate that they lead to substantial improvements. We also show that AdaBoost classifiers are unsuited for this multi-view detection task, and that MPLBoost does not suffer from this insufficiency. The work presented in this chapter was published as [112] and was done jointly with Christian Wojek, who among other things contributed the idea and implementation of MPLBoost.

Chapter 4 – New Features and Insights for Pedestrian Detection
A new feature, color self-similarity (CSS), is introduced in this chapter. It captures long-range similarities of color contributions. By only comparing colors inside the detector window, we successfully circumvent the color constancy problem that one usually has to deal with when using color information. We also introduce a new variant of the optical flow feature from chapter 3 with reduced dimensionality but the same performance. In addition to presenting new features, we highlight pitfalls

[5]This refers to testing for the presence of an object, like testing for a disease – not to testing if the classifier "works".

that exist in common evaluation procedures and can easily distort performance comparisons. Most parts of this chapter were published as [103]. It is joint work with Nikodem Majer, who pushed the idea of using self-similarity features and provided the implementation used in this chapter.

Chapter 5 – Disparity Statistics for Pedestrian Detection
This chapter showcases a combination of static image features, motion features and stereo features. A new stereo-based feature is introduced, which utilizes a relation between disparity and an object's apparent size in the image. Using this feature, the classifier is able to capture scene information like the presence of a ground plane. We also demonstrate that by combining SVMs and boosting classifiers we are able to increase detection performance. The work shown in the chapter was published as [104].

Chapter 6 – Monocular Scene Understanding with Explicit Occlusion Reasoning
In this chapter, we use the pedestrian detector as a component in the 3D scene reasoning framework from Wojek *et al.* [109]. By using multiple detectors trained on partial views of a human and using 3D reasoning, we can infer which parts of the pedestrian should be visible and adjust the score accordingly. This, in combination with a model that encourages temporal consistency of the scene, allows us to detect pedestrians even when they are occluded for extended periods of time. The content of this chapter was published as [111] and it is joint work with Christian Wojek who created the 3D scene reasoning framework with occlusion reasoning. The contribution of the author of this thesis lies in the creation of robust part detectors.

Chapter 7 – Towards a Better Understanding of Feature Combination and Evaluation
This chapter combines and expands on contents from chapters 4 and 5. We revisit the CSS feature from chapter 4 and outline problems with training a classifier on "small" datasets. By using the two-stage training procedure used for stereo classifiers in chapter 5, we are able to use a larger training set and so unlock the discriminative power of CSS, resulting in vast improvements on all surveyed test datasets. We also have a look at local binary patterns and raw color histograms, which we discarded as harmful in chapter 4, and find that raw color histograms are consistently helpful using the enlarged training set. We also highlight further pitfalls during evaluation, especially ones that occur when comparing different approaches.

Chapter 8 – Summary and Conclusion
This chapter concludes this thesis, listing its contributions and detailing a number of interesting open questions that are left for future work.

RELATED WORK 2

Contents

Since pedestrian detection has many useful applications, there has been a lot of previous work on this topic, and many different approaches have been proposed. Because the large amount of publications about pedestrian detection prohibits an exhaustive review of all related work, this chapter focuses on work that has had a significant impact on the field or is especially closely related to the work presented in this thesis. Surveys have also recently been done by Enzweiler and Gavrila [23] and Geronimo *et al.* [45].

Some key differences between pedestrian detection methods are:

Data sources The number and kind of sensors that are employed is an important property of a pedestrian detection system. Using more sensors of course is beneficial for detection performance, however it also restricts the tasks that the method is suited for. While, as will become clear in this thesis, optical flow data is very helpful for pedestrian detection, using it prohibits the use of the detector in e.g. an image retrieval setting. Likewise, using a second camera helps, but it is not always available. Other kinds of data sources are e.g. infrared cameras, laser range data or time-of-flight cameras.

Part-based or global Some methods, especially those aiming at detecting small-scale pedestrians, use a global template for the pedestrian detector, e.g. by trying to match a silhouette or by looking for localized features at specific positions of the detector window. This is usually done in a sliding window framework, where the detector searches for pedestrians by densely sampling detection windows, varying position and scale, or by generating regions of interest in

a preprocessing step (e.g. by background subtraction) that are then classified. Other methods detect lower-level evidence (like body parts, or occurrences of codebook entries) first, and then use this evidence to reason about pedestrian position or pose. These methods usually operate on higher-resolution images of pedestrians, where the high variability of pedestrian appearances prohibit the use of fixed templates.

Tracking Many pedestrian detection systems utilize a tracking module. One reason for this is that tracking can be used to filter out spurious false positives and fill in gaps in detections when a detector misses a pedestrian in one frame, improving detection performance. Another reason for this is that tracking pedestrians enables a system to reason about the future, predicting where a pedestrian will be. This kind of information is invaluable e.g. in driver assistance systems.

Scene understanding Some systems use a model containing a more holistic view of the scene, using e.g. segmentation information or prior knowledge like the tendency of pedestrians to stand on a common ground plane. Such models can reason in 2D (the image) or 3D, and can exist with or without tracking. Like tracking modules, they both improve detection results and offer an additional kind of information for applications.

Since those points and other distinctions like employed features and classifiers or the presence of occlusion reasoning are mostly orthogonal, it is hard to present the related work in a hierarchical fashion because each work would have to be mentioned in multiple categories, e.g. many systems using a stereo camera also have tracking and scene reasoning modules. Because of this, publications will be mainly listed in the section they fit most.

2.1 TEMPLATE-BASED PEDESTRIAN DETECTION

One popular way to build a pedestrian detector, which is also the method used in the main part of this thesis, is to represent candidate bounding boxes by a set of features computed from the contents of the box, sometimes with additional context. Statistical learning methods [7, 36, 37, 79] are then applied using pedestrian and non-pedestrian training data, aiming to find a general rule to distinguish pedestrian from non-pedestrian feature vectors. This works best for low-resolution images, because the variability of human appearance is less evident in small scales. One of the first works on pedestrian detection using this method is from Oren *et al.* [69] (later extended in Papageorgiou and Poggio [71]), who build an over-complete feature representation of the detection window using horizontal, vertical and corner Haar wavelets. They train a polynomial support vector machine (SVM) to distinguish pedestrians from non-pedestrian crops, showing good performance on the dataset they introduced, the MIT pedestrian database.

Another early work on pedestrian detection is Gavrila and Philomin [43], who use a variant of the distance transform to match shape templates to images in real-time. They employ a template hierarchy and match templates in a coarse-to-fine fashion, exploiting the fact that, while pedestrian shapes are quite different when seen from multiple viewpoints, on a coarser scale they are quite similar.

A very effective approach was chosen by Dalal and Triggs [11], who introduce the histograms of oriented gradients (HOG) feature. The feature shares properties with SIFT [61, 62] and shape context [6]. It consists of a dense grid of gradient histograms with trilinear interpolation (between histogram bins and between histograms) and local normalization. With HOG and a linear or kernel SVM, they obtain essentially perfect results on the MIT pedestrian database. They introduce a new database, INRIA PERSON, containing people in various poses, and also show that their methods performs best on this dataset. This combination – HOG and SVM – is very successful for many object detection tasks and forms the basis for many current state-of-the-art approaches.

Zhu *et al.* [123] combine the feature of Dalal and Triggs [11] with the cascade approach of Viola *et al.* [100]. They learn many SVMs, each on one HOG block. However, unlike in [11], those blocks have varying sizes, positions and aspect ratios. They are able to obtain a big reduction in runtime speed without sacrificing much performance.

Another extension to [11] is Wang *et al.* [105], who combine the HOG feature with a cell-structured instance of local binary patterns [68], attempting to capture texture information. Local binary patterns are invariant to strict monotonic changes in intensity, making this feature robust against changes in lighting. They also add occlusion handling to the detector by trying to infer which blocks are visible from the response of the linear SVM. They divide the SVM score into contributions from each block, and check if the contribution of the blocks is similar inside the detector window. If their system detects that SVM scores are inconsistent across the detector window, they employ a partial detector that is trained on the portion of the detector window that is assumed to be visible.

Maji *et al.* [63] also extend [11] with a multiresolution pyramid feature and a fast approximation to the histogram intersection kernel, which exploits the fact that the intersection kernel is additive and so the resulting decision function can be independently computed (and approximated) for each dimension[1]. Vedaldi and Zisserman [98] also exploit this property, however they derive the generation of explicit feature maps for the general class of additive kernels, showcasing noticeable improvement on the pedestrian detection task by using the χ^2-kernel.

Dollár *et al.* [17] employ feature mining to select good binary features for use in a boosting classifier. They study several strategies to select subsets of the feature space, including random sampling, sampling for features that allow a good separability of the data and features that are complementary to the already selected features

[1]Their evaluation procedure had a flaw where over-fitting on artifacts in the positive training samples occurred, see [19] for details and updated results.

(selected via clustering using a metric in the binary feature space), and find that complementary features work best.

Sabzmeydani and Mori [76] use AdaBoost [35] on low-level gradient features to select informative higher-level features, continuing to learn an AdaBoost classifier on those learned features. The aim of this is to circumvent the problem that, in order to make features robust and general, a lot of information is discarded during the feature computation stage[1].

Tuzel *et al.* [92, 93] use covariance matrices as features, computed over various regions in the detection window using low-level features like spatial position, intensity and gradients as input. In order to obtain a classifier suited for this feature space, they extend LogitBoost [37] to work on Riemannian manifolds, showing a substantial improvement over [11] on INRIA PERSON. Covariance matrices also allow for a very natural treatment of the combination of different localized features.

Dollár *et al.* [16] use sums over rectangular regions in a number of low-level "feature channels" (like channels from the *CIELUV* color space, gradients, vote strengths for different HOG bins) and train a boosted classifier on those features. The sums are computed using the integral image technique, resulting in fast feature computation times while producing state-of-the-art results. This is later extended in Dollár *et al.* [15] using multiple optimizations and approximations like interpolating gradient votes between different scales, which results in near real time processing speeds on a single CPU without sacrificing detection performance.

Wu and Nevatia [116] integrate runtime considerations into a boosting framework. They evaluate weak classifiers in descending order with regard to classification power, normalized by the computational cost. This means that weak classifiers that are computationally cheap or highly predictive are preferred. If the classifier is confident enough in its assessment, no further weak classifier is evaluated. This allows them to combine multiple features with different computational requirements, optimizing the trade-off between discriminative power and runtime.

Schwartz *et al.* [80] augment HOG with other features, reaching a total feature dimensionality of over 170 000. Since learning a classifier in this feature space, especially with the low amount of training data available, is intractable, they apply partial least squares (PLS) as a dimensionality reduction mechanism, using quadratic discriminative analysis on the projected features as a classifier.

2.2 PART-BASED DETECTION

Another way to detect the presence of a human in an image is detecting lower-level evidence (like body parts) first, and combine the collected evidence to reason about the higher-level structure – the pedestrian. An example for that is the part-based model of Mohan *et al.* [67], which extends the work of Papageorgiou and Poggio [71] (from the previous section). In this work, head, arm and leg models are separately trained. For each part, the best response is collected (under the constraint that the

estimated configuration is feasible) and used as input for a quadratic SVM producing the final confidence score. Mikolajczyk *et al.* [66] also model humans as a flexible assembly of parts, using AdaBoost on co-occurrence of orientation features as a body part detector. They later group part detections in a probabilistic way to generate human hypotheses.

The implicit shape model (ISM) by Leibe and Schiele [57] also aggregates local evidence to form object hypotheses. A visual codebook of image patches on interest points is constructed. The spatial occurrence distribution of codebook entries in relation to the object centers in *x-y-scale* space is learned. At test-time, codebook entries vote for object centers. Hypotheses are then formed by performing mean shift in the voting space. Leibe *et al.* [58] combine detections of the ISM with a verification stage using segmentation and chamfer matching. The verification stage helps the problem that the ISM does not put any restriction on the assignment of local evidence to object hypotheses, leading to e.g. humans with evidence for 3 legs. Since this disagrees with the segmentation and chamfer matching cues, the verification stage can improve the assignment and help detecting pedestrians in crowded scenes.

Seemann *et al.* [81], instead of learning an individual occurrence distribution for each codebook entry as in the original ISM, learn a joint occurrence distribution for all codebook entries, enabling their system to handle codebook entries with different importance, making learning from few samples more robust and resulting in scores from different models becoming comparable. They also investigate instance-specific models that can be used e.g. to re-identify pedestrians after they became temporarily invisible because of occlusion.

Gall *et al.* [39] (based on [38]), like [57] use a framework based on the generalized hough transform, which detects parts of an object and then assembles the parts together in a probabilistic framework. However, instead of learning a codebook in an unsupervised way, they discriminatively learn the features that vote for object positions. They employ random forests, which are able to cope with multi-view/multi-aspect classes, and suitable for adapting them online to specific object instances.

Maji and Malik [64] also combine an ISM-like framework with discriminative learning. They use a formulation which, given the activations of the codebook entries for a number of positive and negative training samples, learn weights for the codebook entries, optimizing the margin. They combine this with a verification step that scans the environment of generated hypotheses with an intersection kernel SVM.

Tran and Forsyth [91] argue that global models like [11] are inefficient and employ structure learning to learn the configuration of human body parts. Their method deals better with unusual articulations that are not well-represented in the training set and is able to cope with misalignment of bounding boxes. They also point at problems with the evaluation protocol for FPPW introduced in [11], an observation also made in [18, 19].

Andriluka *et al.* [1] combine key point detectors and a probabilistic voting scheme with a model for the human walking cycle, enforcing temporal consistency of detections across multiple frames in a tracking-by-detection framework. In Andriluka *et al.* [2] it is shown that by combining a powerful but simple generative model – the pictorial structures model [32] – with strong discriminative part detectors – AdaBoost on dense shape context features [6] – one is able to obtain very good results in pedestrian detection as well as upper- and full-body pose estimation. Andriluka *et al.* [3] use the strong local evidence from [2] and the walking cycle model of [1] as a basis to build a multi-stage system to predict the 3D body pose from a single camera.

Dollár *et al.* [14] use multiple-instance learning to iteratively add discriminative classifiers on components in a boosting framework. This is done in a weakly supervised fashion – there are only bounding box annotations for the whole object, not the components. A drawback of this method is that it results in many shallow cascades that are computed in parallel instead of a single deep cascade as in [100], resulting in increased computation costs.

Shashua *et al.* [84] use a multistage classifier where SVMs are trained on gradient statistics of subregions of pedestrian windows. The SVMs are trained multiple times, each on a (manually selected) cluster of the training data corresponding to a specific viewpoint. The scores of the SVMs are combined via AdaBoost. Their detector is part of a system that preselects regions of interest based on e.g. geometric constraints and texture, and filters detection results based on temporal consistency and gait.

Lin and Davis [60] decompose the shape model into parts, constructing a part-template tree that is matched in a hierarchical way, with feature descriptors that are adapted to body poses, leading to a detector that is tolerant to pose variations. The HOG-like features are sampled at the borders of the pose edges, which are the regions that are most discriminative [11]. In addition to detecting humans, the matched part templates provide a segmentation that is very accurate for "standard" poses of humans.

Schnitzspan *et al.* [77] use a conditional random field with an extension to structure learning. They automatically discover relevant long-distance feature couplings in an hierarchical model, demonstrating the beneficial effect over a fixed structure. In [78], they extend this model to automatically discover meaningful parts. Part labels are treated as latent variables, using pairwise potentials between the part nodes to model the spatial layout of parts. The model is able to cope with articulation and viewpoint changes as well as occlusion.

Felzenszwalb *et al.* [33] combine the successful global feature vector by Dalal and Triggs [11] with a part-based approach. In their popular discriminative part-based model (DPM), multiple part filters consisting of higher-resolution HOG templates are learned, treating part locations as latent variables. The model optimizes a combination of image evidence (the response of global and part filters) and a deformation cost, penalizing deviations of part positions from their mean position. They later extend this in Felzenszwalb *et al.* [34] with a mixture model, which can be

used e.g. for multi-view detection, where the mixture component is an additional latent variable. They also replace HOG by a PCA-inspired lower-dimensional variant, without affecting detection accuracy adversely, and formalize their training procedure.

Park *et al.* [72] counteract one drawback of the DPM model (that it does not work well on small scales) by adding scale as another latent variable, using models specialized for the respective resolution ranges. At low resolutions, only a fixed HOG-like template is used. For larger candidate windows, the high-resolution part templates are activated. They also implement a simple scene model which puts the lower y-coordinate in relation to the size of the bounding box, implying a common ground plane between the detections.

One approach that does not explicitly model parts but also does not rely on a global template is presented by Lampert *et al.* [55, 56]. They propose a framework using a branch-and-bound technique to make object detection more efficient. Their method relies on a feature/classifier combination having a (tight) upper bound for the classification score of a set of detection windows, resulting in far less classifier evaluations. They use a linear SVM with bag-of-words features, for which this upper bound is easy to derive, however this feature has no localization information and so the classifier is not as powerful.

2.3 BENCHMARKS AND EVALUATIONS

There are different datasets and different evaluation protocols used for testing object detectors. As not every detector that is proposed is evaluated with the same setting initially, it is hard to draw meaningful comparisons between approaches. Therefore it is helpful to perform benchmarks of existing detectors in a consistent test setting, and there have been multiple recent publications covering this topic.

Wojek and Schiele [110] evaluate different static image features like HOG, Haar wavelets and dense shape context in combination with AdaBoost and support vector machines as classifiers. They also explore the possibilities of combining multiple features to improve detection.

A more extensive benchmark for pedestrian detection was done by Dollár *et al.* [18]. They evaluate multiple algorithms on a new dataset, the Caltech pedestrians database, which is the biggest pedestrian database to date. Several problems with per-window evaluation are uncovered, especially that algorithms that can be ranked well in a per-window evaluation setting can perform poorly in a per-image evaluation, which is a better measure of the performance in the task usually defined as pedestrian detection ("find the bounding box of every pedestrian in this image"). They also perform an evaluation of the effect of recurring problems in pedestrian detection like occlusion, unusual aspect ratios and small scale. Dollár *et al.* [19] extends this evaluation, including more algorithms, a refined evaluation criterion and a more detailed analysis of occlusion patterns.

Enzweiler and Gavrila [23] perform a series of experiments, testing various combinations of features and classifiers suitable for detection from an on-board camera under a processing time constraint. They find that for higher resolutions and slower processing speeds, HOG in combination with a linear SVM performs best among the features they surveyed, however at low resolutions and fast processing speeds a boosting cascade on Haar features is preferable.

2.4 USE OF MOTION FEATURES

Given that the movement of objects is an important cue for the visual perception of humans, the inclusion of motion information into a detector system seems like a natural step to take. However, how to encode this information is a question that is not answered easily, and so different approaches have been proposed and tested.

Viola *et al.* [100, 101] find that their successful face detector [99], which employs a cascade of boosting classifiers using wavelet features, does not perform as well on pedestrians, and extend their feature representation with wavelets on temporal difference images with spatial shifts, substantially improving performance. This works well in a surveillance setting where the cameras are static, however this method will not work well in the case of a turning or moving camera.

The work of [11] is extended with motion features in Dalal *et al.* [12], where they evaluate different motion descriptors, including the one by [100]. They introduce various flow-based descriptors falling into two categories, motion boundary histograms and internal motion histograms. Motion boundary histograms (MBH) are closely related to HOG, they capture short-range gradients in the horizontal and vertical components of optical flow. Internal motion histograms (IMH) capture differences of flow vectors over a longer range. They find that MBH works best on its own, but in combination with HOG IMH works better, probably because its lower correlation with HOG – flow edges tend to coincide with image edges. However, the flow-based descriptors worked better than the difference wavelets of [100]. Training and testing of the classifier was done on a dataset consisting of clips from feature films, which means frequent camera turns (but camera ego-motion is rare).

Enzweiler *et al.* [24] use motion as a cue for their region-of-interest generator. They compute optical flow, adjusting it by the flow field that a ground plane would generate at the current speed (estimated from other sensors). The resulting flow field captures objects that are standing out from the ground plane (or are moving independently). At the later stages (detection, classification and tracking) motion information is not used.

Sharma and Davis [83] work in a surveillance setting (a static camera viewing the scene from above), employing a Markov random field for simultaneous detection and segmentation of pedestrians. The segmentation output does not require segmented training examples, it is inferred automatically. They use both appearance and motion cues for their system, reasoning that contour fragments sharing the same motion

behaviour are likely to belong to the same object.

2.5 SELF-SIMILARITY

One of the cues used in this thesis is self-similarity[2]. Self-similarity features have the beneficial property that they usually generalize better than the low-level features that they are computed on, because things that look similar in one database tend to look similar in other databases, even if there is significant dataset bias e.g. in terms of color distribution or noise level. Probably the most prominent work on self-similarity in computer vision is Shechtman and Irani [85], who propose a densely sampled local self-similarity descriptor, using it for comparisons across images and videos. A patch of the image is compared with subregions of a larger, surrounding patch, storing the results in a log-polar grid. Showcased settings are sketch-based and image-based image retrieval, and action recognition (using a spatio-temporal variant of the descriptor).

Stauffer and Grimson [89] propose a representation similar to our CSS descriptor (chapter 4), where color similarity is computed at the pixel level, assuming a Gaussian conditional color distribution. Using their self-similarity pedestrian template, they automatically learn a segmentation of detection windows into foreground/background, pants, shirt, and other parts, using it e.g. for database queries ("find pedestrians with a shirt like this one").

Deselaers and Ferrari [13] introduce a global self-similarity descriptor for image classification and object detection, presenting ways to compute the global self-similarity descriptor efficiently in runtime and space requirements via an image-specific codebook.

Junejo *et al.* [52] observe that, while actions look differently when observed from different viewpoints, self-similarity patterns across time are stable across viewpoints. They track points on the body to compute a trajectory-based self similarity descriptor. They also show that performance improves when self-similarity measures based on different input features are combined.

Vedaldi *et al.* [96] use a multi-stage multiple kernel learning method for object detection. One of the input features they use – in addition to features like histograms of gradient directions and bag of visual words – is the self-similarity descriptor of [85] . In contrast to [85], they quantize the descriptor by using a codebook of 300 visual words.

Ott and Everingham [70] utilize the idea of instance-specific color for pedestrian detection, using an implicit segmentation of HOG cells into foreground and background, based on the color distribution. The segmentation is used as input for a HOG-like feature and combined with HOG to improve pedestrian detection results.

[2] In computer vision, self-similarity does not refer to the property of being the same or similar to a part of itself (e.g. in the case of fractals), but of different parts being similar to each other.

2.6 STEREO AND DEPTH FEATURES

A second camera enables a vision system to estimate a depth map. This depth map can be used in multiple ways. For example, many systems in this section use it to reason about the scene like estimating a ground plane or enforcing a height prior on pedestrians. However, there is also the possibility of directly encoding depth information into the feature vector, which is what the methods presented in this section are doing.

Rohrbach *et al.* [75] build a HOG-like feature, using the depth field generated by a dense stereo matcher as input, reporting a substantial reduction in false positive rates at a given sensitivity. They also experiment with multiple ways to combine the HOG and stereo-HOG feature spaces, including concatenating the feature vector and training separate classifiers on each feature space, and combining them with a fixed or learned rule. Enzweiler *et al.* [21] combine HOG-like features on intensity, (the x-component of) optical flow and depth with a part-based model and a prior on pedestrian shape into a system that reasons about partial occlusion by segmenting the optical flow and stereo fields. The weights for the partial detectors are adapted based on the estimated occlusion mask.

Rapus *et al.* [74] utilize a low-resolution (64×8 pixels) 3D camera (time-of-flight principle), and extract multiple features, including gradients and Fourier coefficients, from intensity and depth to detect pedestrians after a preprocessing step to enforce a common ground plane.

Hattori *et al.* [47], like [21], build a HOG-like feature on the depth field, however they also model co-occurrence statistics between histogram bins (like [106] did for HOG), obtaining better characteristics than with using the HOG-like feature alone. They also utilize the second camera by classifying the detector window in both cameras, combining the score afterwards.

2.7 TRACKING

As mentioned in the introduction, tracking mainly serves two purposes. It stabilizes detections, using a prior on temporal consistency to suppress false positives and fill in missing detections, and enables the system to reason about the future. Tracking modules are often (but not necessarily) coupled or integrated with scene understanding modules. One example for such an integrated system is Ess *et al.* [25, 26, 27, 28], who build an integrated pedestrian detection and tracking system. They fuse information from depth estimation, object detection, visual odometry and tracking modules, including an failure-resistant way for handling the feedback loops that arise in such a setting, e.g. because the positions of pedestrian detections are used to estimate the ground plane (together with other cues like the depth map), but the depth map is also used to reason about pedestrian detections. Stereo is also used to enforce a prior on human height and to verify that detections have a consistent

depth. They show robust tracking results in difficult scenes recorded from a moving platform.

Gavrila and Munder [42] propose a pipeline of Chamfer matching and several image based verification steps for a stereo camera setup. They use stereo information in two ways: first, they identify regions of interests in the disparity maps; after the pedestrian detection step, hypotheses are verified by cross correlation between the two images – if there is no object at the estimated disparity level, the correlation measure is low and the hypothesis is rejected. After the detection stages, tracking is applied to suppress spurious false positives and fill in gaps in detections.

Choi and Savarese [8] tackle the problem of monocular tracking by employing a 3D model which handles track interactions like repulsion (persons can not occupy the same space in 3D) and attraction (groups are likely to stay together). The camera parameters that are needed to reason about the 3D world with input from the image plane are automatically inferred.

Isard and MacCormick [49] employ blob-tracking in a surveillance setting with a static background. They propose a scene likelihood which is comparable across configurations with different numbers of objects. They use particle filtering with a Bayesian blob tracker that handles a varying number of objects.

2.8 OCCLUSION REASONING

A common problem for many detectors is occlusion or missing evidence in general, and there has been a lot of interest in remedying this issue. The already mentioned method of Wang *et al.* [105] uses local SVM scores to infer occlusion patterns, Enzweiler *et al.* [21] use segmentation on optical flow and stereo together with a prior on human shape to detect occlusion. In the case of assumed occlusion, both methods apply a detector tailored for this occlusion pattern.

There are also methods that reason about occlusion at a later stage. For example, Zhang *et al.* [122] use a network flow optimization technique with an explicit occlusion model for tracking. They handle occlusion in an iterative way: First, the method is run without occlusion reasoning. From this, hypotheses about occluded detections are generated and the method is iteratively repeated.

Lin *et al.* [59] adapt the boosting cascade of [99] to be able to handle occlusion. If a window is classified as negative by the full-object cascade and the weak classifier evidence is consistent with an expected occlusion pattern, a cascade specialized for this occlusion pattern is employed. They also employ reinforcement learning to reduce false positive rates and adapt the boosting scheme in order to prevent bad performance caused by overfitting on outliers.

Ess *et al.* [29] explicitly handle occlusion by combining information from an occupancy map containing information about static (non-pedestrian) obstacles with estimated pedestrian positions into an occlusion map specifying if a pedestrian standing on a given point of the ground plane should be visible. Trajectories moving

across occluded territory are not discarded, enabling their system to track through occlusion.

Vedaldi and Zisserman [97] use structured output regression for object detection. Their method is able to handle flexible parts to some extend by rearranging HOG blocks, also they able to handle missing evidence caused by truncation at the image border (where the parts of the bounding box that are invisible are fixed, given the object bounding box).

Gao *et al.* [40] use a similar model, however they also model occlusion that is not caused by truncation at image borders. They use latent variables to model occlusion, determining if the "object" or "occluder" model should be used for each cell. A smoothness term is responsible for ensuring spatially coherent occlusion maps. Their system also encourages global consistency of occlusion maps, modelling object-object occlusion.

Wu and Nevatia [113, 115] introduce edgelet features, which capture local gradient configurations like lines and arcs. They use a multi-part configuration (full-body, head, torso and legs), learning AdaBoost classifiers for each part, and building a scene model for object-object occlusion, assuming that pedestrians stand on a fixed ground plane, and the camera views the scene from above. This allows the detector to explain missing part evidence if the part is expected to be occluded from another pedestrian. They later use edgelet features in Wu and Nevatia [114], where they introduce cluster boosted trees, an extension to real-valued AdaBoost where they cluster the instances based on discriminative features, resulting in a tree-structured classifier that is able to cope with multiple viewpoints. Wu and Nevatia [117], Wu *et al.* [118] use edgelet features and cluster boosted trees in a more fine-grained part-based approach, where parts are arranged in a tree structure (with the full-body detector being the root node) and features are shared between parts and their ancestors.

Xing *et al.* [119] use particle filter tracking with multiple partial detectors. They use a full-body detector and a head-torso and a head-shoulder detector, arguing that those parts undergo the least variation and are therefore most easily to detect. Also, in a surveillance setting, it is the lower part of the body that is occluded most often.

Winn and Shotton [108], like [78], employ a CRF with localized latent part variables, modelling self-occlusion, occlusions caused by instances of the same class that is occluded and instances of other classes separately. Among the cues for occlusion caused by the same class are labels that belong to the same class but are spatially inconsistent. They use the CRF for simultaneous segmentation and detection.

Sigal and Black [87] model self-occlusion in a pose estimation setting. By explicitly modeling occlusion, they try to overcome a problem that is common when employing pictorial structures model, that the same region of the image is assigned multiple body parts (e.g. both legs, implying one leg occludes the other) while the actual body part location is left unexplained because it has a lower part likelihood.

2.9 DATASETS AND TRAINING

Another recurring problem in pedestrian detection is that usual settings break the fundamental assumption that statistical learning methods like boosting or SVMs are based on: That the test samples are *independently* sampled from the *same* distribution as the training samples. Feature design usually aims at alleviating this issue, e.g. by making the feature invariant to lighting changes. Still, how to collect a database to train a real-world pedestrian detection system is an open question, leading to the introduction of new datasets and research on what makes a training set "good".

Torralba and Efros [90] have a closer look at the history of training and testing databases, identifying that the usual cause for introducing a new dataset is a perceived bias in the established datasets. However, those new datasets are not bias-free either, which is demonstrated by learning a classifier that can tell to which dataset a specific instance belongs (which should not be possible for "realistic" images). They also study the "worth" of training samples by studying how many training samples from set A are needed to replace one training sample of set B if one wants to keep the performance constant.

Enzweiler and Gavrila [22] address the problem that training databases usually poorly cover the high variability of human appearance. They use a generative model to expand a training database with many additional training samples gained by modifying existing samples, varying background, shape and texture. A discriminative classifier is employed on the enriched data set. Active learning is used to only add informative samples from the generative process – samples that the discriminative classifier is currently unsure about.

Going one step further, Marin *et al.* [65] explore the possibilities of training discriminative detectors on training examples generated by a computer game engine. Since the samples are computer-generated, pixel-level annotations are easy to produce. Using HOG features, they are able to use virtual training samples to reach detector performance comparable to a detector trained on the Daimler training set.

Pishchulin *et al.* [73] use MovieReshape [51] to generate training data for the pictorial structures model of [2] and a sliding window HOG detector from few input models. Using only 6 input models, seen in multiple poses and from multiple viewpoints and varying the height of the pedestrians they were able to get good detection performance, significantly better than they got from training on the data used in [65].

2.10 RELATION TO THE PRESENT WORK

Undoubtedly the most influence on this work came from the work by Dalal et al. [10–12]. Their HOG feature is a main component of all our pedestrian detectors, and the techniques leading to its success have been integrated into our novel features, where applicable. While they were not able to get their motion feature to work in a

full-image setting, the work presented in this thesis does so by eliminating a flaw in the non-maximum suppression scheme and taking care that the flow fields seen at training time get the same treatment as those at test time. To the best of our knowledge, our system is still one of very few systems which are able to use motion information in an on-board setting. Similar approaches are Enzweiler *et al.* [24] who use motion information to generate regions of interest and Enzweiler *et al.* [21] who do use motion features, but contrary to our system only in a classification setting (where only samples generated from a preprocessing step have to be classified).

This work also benefits from advancements on the classifier side. The approximation technique from Maji *et al.* [63] (which was later generalized by Vedaldi and Zisserman [98]) allowed us to incorporate the intersection kernel for SVMs, which was previously infeasible because of runtime constraints. This gives the detector presented in this thesis an edge over systems like those by Dalal [10] which used a linear SVM for the full-image setting for speed reasons. MPLBoost from Babenko *et al.* [4] (independently derived as MCBoost by Kim and Cipolla [54]) provided us with a boosting classifier which we discovered to be able to handle a multi-view problem (as arises in on-board pedestrian detection) when using decision stumps as base classifiers, unlike AdaBoost. This means that the detectors presented in this thesis have an edge over detectors like those used by Wojek and Schiele [110].

Of course, our new features also benefited from influence by related work. The inspirational publications for the CSS feature were Shechtman and Irani [85] and Stauffer and Grimson [89], who highlighted the use of self-similarity in detection and the use of color information. In contrast to those works, we compute long-range similarity measures within the detector window, enabling the feature to capture the global structure of a pedestrian. One feature (StereoHOG) from chapter 5 is a close relative to the stereo-based HOG from Rohrbach *et al.* [75]. The other new stereo feature presented in this thesis is, as far as we can tell, unique in its approach, and gives consistently results that are as good or (more frequently) better than StereoHOG. Enzweiler *et al.* [21] also combine appearance, motion and stereo features, but as already mentioned they focus on the classification task and use the same stereo feature as [75].

The direction that Felzenszwalb *et al.* [34] (extended by Park *et al.* [72]) and Andriluka *et al.* [2] take is a different one from the one present in this thesis. They build on relatively "simple" features (HOG and dense shape context respectively) and work on a powerful model on top of those features, allowing for flexible representation of pedestrians and leading to good detection performance. In contrast to this, in this thesis we focus on creating a good feature set while keeping the architecture of an "unflexible" feature vector which is used as input for a classifier fixed. It is very likely that a combination of those approaches will result in noticeably better performance than is reachable now.

The need to carefully select the evaluation procedures, which is highlighted in chapters 4 and 7, is also shown in Wojek and Schiele [110] and Dollár *et al.* [18], who demonstrate that using FPPW as a measure for comparing algorithms can lead

to flawed conclusions. During the analysis of the effects of retraining in chapter 4, Felzenszwalb *et al.* [34] was useful as it provides a proof of convergence for the retraining procedure using SVMs.

The problem of occlusion, which we tackle in chapter 6, is well-known. There have been multiple approaches to this problem. Most approaches (e.g. Zhang *et al.* [122] or Ess *et al.* [29]) do not aim to detect occluded pedestrians, but merely to track them through occlusions. Wang *et al.* [105] tried to detect occlusion by noticing spatially correlated irregularities in local contributions to the SVM score, and employing specialized detectors when they detected occlusion. This worked reasonably well on their synthetic dataset, however on a real-world dataset as [18] it did not work as well. Their approach is similar to Lin *et al.* [59], which employ a boosting cascade and look for patterns in the weak classifier responses to infer occlusion. The closest relation to our work has Gao *et al.* [40], who develop a globally consistent model of object-object occlusions, implicitly segmenting the detector window into foreground and background. Contrary to them, who are only able to show a significant improvement for cars, we are able to show the beneficial effect of our occlusion handling for pedestrians.

MULTI-CUE ONBOARD PEDESTRIAN DETECTION 3

Contents

3.1 INTRODUCTION

While psychologists and neuroscientists argue that motion is an important cue for human perception [46] to detect people and other moving objects, only few computer vision object detectors (e.g. [12, 100]) exploit this fact. Viola *et al.* [100] showed improved detection performance but for static cameras only, because they rely on temporal differences with a fixed spatial shift. It is unclear how to transfer their results to on-board sequences.

In contrast, Dalal *et al.* [12] proposed motion features that are based on optical flow. While they showed improved performance using the FPPW evaluation criterion (False Positives per Window) they were unable to outperform their own static HOG feature [11] in a complete detector setting [10]. Also, while their representation should be in principle usable for on-board sequences, they have only been tested on a motion picture database, where the camera is mostly static (there is some camera pan, but scenes where the camera is moving forward, as is typical in on-board sequences, are rare). We will show in this chapter that the feature is indeed suitable for on-board sequences and detail how to make it work in a full-image detector setting.

The second avenue we follow in this chapter is to incorporate multiple and

complementary features for detection. While Varma and Ray [95] convincingly showed that multiple features improve performance for image classification, for detection only few approaches exploit this fact [42, 110, 116].

The third avenue of this chapter is related to the classifier choice. Popular classifiers are SVMs [11, 33, 60, 63, 84] or boosting [14, 76, 102, 114]. However, the large intra-class variability of pedestrians seems to require a more careful design of the classifier framework. The choice of the classifier framework is closely connected to the data's distribution. In particular, multiple viewpoints give rise to distributions which are hard to learn.

While kernel SVMs are a popular choice for those cases when SVMs are used as classifiers, they become practically infeasible for the task of detection when the number of support vectors becomes large. AdaBoost on the other hard suffers from its inability to learn distributions which are arranged in a XOR layout when decision stumps are chosen as weak classifiers.

Wu&Nevatia [114] remedy this issue by learning a tree structured classifier, Lin&Davis [60] use a handcrafted hierarchy, while Seemann et al. [82] propose multi-articulation learning. Gavrila [41] proposes a tree-structured Bayesian approach that builds on offline clustering of pedestrian shapes.

What is common to these approaches is that they treat the problem of data partitioning and classifier learning separately. In this chapter however we address this problem in a more principled way by using the MPLBoost classifier [4] that simultaneously learns the data partitions and a strong classifier for each partition. Multiple strong AdaBoost classifiers are learned jointly in this framework, each one focusing on a subpart of the data. Moreover, clusters of similar data are determined automatically based on discriminative features and thus no preprocessing such as clustering is required.

In this chapter, we show that motion cues provide a valuable feature, even when the sequence is recorded from a moving platform. We also show that MPLBoost and histogram intersection kernel SVMs can successfully learn a multi-viewpoint pedestrian detector and often outperform linear SVMs. In order to do this, a new realistic and publicly available onboard dataset (TUD-Brussels) containing multi-viewpoint data is introduced. It is accompanied by one of the first training datasets (TUD-MotionPairs) containing image pairs which allow to extract and train from motion features. These two datasets will enable comparison of different approaches based on motion. In addition to the main points of this chapter we discuss several important algorithmic details that are often neglected and overlooked.

3.2 FEATURES AND CLASSIFIERS

In the following subsections we will discuss the features (section 3.2.1) and classifiers (section 3.2.2) which we deploy in a sliding window framework.

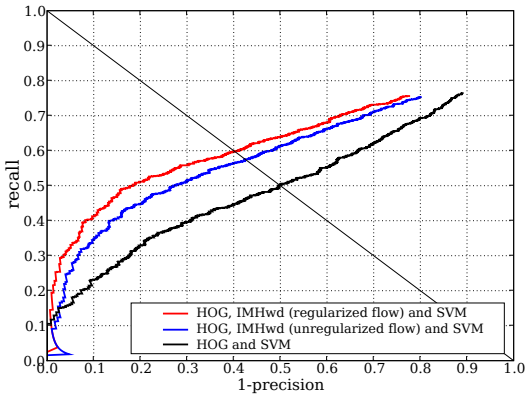


Figure 3.1: Performance for different flow algorithms – using the regularized flow algorithm by Zach et al. [121] works better than using the unregularized one detailed in [10].

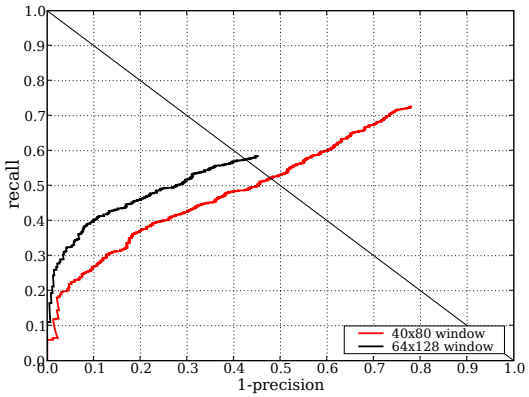


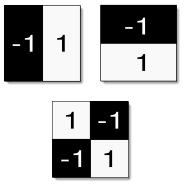
Figure 3.2: Performance drops when using a smaller detection window. Because of this performance drop, we scale the images up instead of reducing the window size.

3.2.1 Features

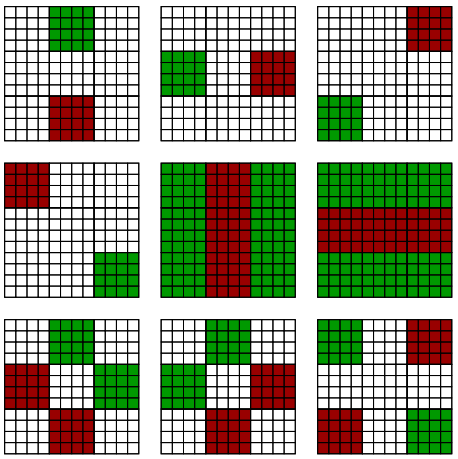
A wide range of features has been proposed for pedestrian detection. Here, we focus on three successful features containing complementary information (see [110] for a wider range of features). While HOG features encode high frequency gradient information, Haar wavelets encode lower frequency changes in the color channels. Oriented Histograms of Flow features exploit optical flow and thus a complementary cue.

HOG Histograms of oriented gradients have originally been proposed by Dalal&Triggs [11]. The bounding box is divided into 8×8 pixel *cells* containing histograms of oriented gradients. Image derivatives are computed as simple center differences in x- and y direction. Magnitude is collected in histograms weighted with respect to the spatial location within the cells and with respect to the gradient orientation. 2×2 cells constitute a *block* which is the neighborhood to perform normalization. For people detection L^2-norm with an additional hysteresis step (that prevents a single histogram entry from dominating the feature vector) performs best. Additionally, blocks overlap by 50% and thus cells are represented multiple times, normalized with respect to different neighborhoods. To the right, an example HOG pedestrian model is visualized, showing dominant gradient orientations for each block.

Haar Haar wavelets have been introduced by Papageorgiou&Poggio [71] for people detection. Those provide an overcomplete representation using features at the scale of 32 and 16 pixels. Similarly to HOG blocks, wavelets overlap by 75%. As proposed we use the absolute responses of horizontal, vertical and diagonal wavelet types.



Oriented Histograms of Flow The motion feature we use throughout this chapter is the Internal Motion Histogram wavelet difference (IMHwd) descriptor described by Dalal et al. in [10, 12]. The descriptor combines 9 bins per histogram on 8×8 pixel cells, with interpolation only for histogram bins. It is computed by applying wavelet-like operators on a 3×3 cell grid, letting pixel-wise differences of flow vectors vote into histogram bins. We use IMHwd due to its consistently better performance in previous experiments compared to other proposed descriptors. The flow field is computed using the TV-L^1 algorithm by Zach et al. [121], which provides regularization while allowing for discontinuities in the flow field. We also conducted experiments with the unregularized optical flow algorithm described in [10], resulting in a slight loss of performance compared to the algorithm by Zach et al. [121] (cf. figure 3.1). Contrary to [10], where the motion descriptors did not lead to an improvement on the image level evaluation, we obtained a substantial increase of performance using motion information for multi-view data (cf. figure 3.1 and second row of figure 3.9 and figure 3.10). Probable reasons for this are that contrary to [10], we compute the optical flow for the training samples on full images instead of crops, which is particularly important for the regularized TV-L^1 flow (because it suffers from boundary artifacts otherwise, resulting in different feature statistics for training and test sets). Also, the motion-enhanced detector's performance seems to be more sensitive to the choice of the non-maximum suppression method (cf. section 3.3.2 and figure 3.10f).



Feature combination In the experiments reported below we analyze various combinations of the above features. To combine features we L^2-normalize each cue-component and concatenate all subvectors. The concatenated feature vector is the input for the classifier algorithm.

3.2.2 Classifiers

The second major component for sliding window based detection systems is the employed classifier. Most popular choices are linear SVMs and AdaBoost. As dis-

cussed before these are not perfectly suited because of the high intra-class variability of humans e.g. caused by multiple viewpoints and appearance differences. In this chapter we therefore explore the applicability of MPLBoost that learns data clusters and strong classifiers for these clusters simultaneously.

SVM

Linear SVMs learn the hyperplane that optimally separates pedestrians from background in a high-dimensional feature space. Extensions to kernel SVMs are possible, allowing to transfer the data to a higher and potentially infinity dimensional representation as for RBF kernels. For detection however, kernel SVMs are rarely used due to higher computational load. One remarkable exception is Maji et al. [63] who approximate the histogram intersection kernel for faster execution. Their proposed approximation is used in our experiments as well.

AdaBoost

Contrary to SVMs, boosting algorithms [37] optimize the classification error on the training samples iteratively. Each round a weak classifier is chosen in order to minimize the weighted training error. The weighted sum of all weak classifiers forms the final strong classifier. A typical choice for weak learners, which are required to do better than chance, are decision tree stumps operating on a single dimension of the feature vector. In this thesis, we use AdaBoost as formulated by Viola and Jones [102].

MPLBoost

MPLBoost by Babenko *et al.* [4] (independently formulated as MCBoost by Kim and Cipolla [54]) is a recently proposed extension to AdaBoost. While AdaBoost fails to learn a classifier where positive samples appear in multiple clusters arranged in a XOR-like layout, MPLBoost successfully manages this learning problem. This is achieved by simultaneously learning K strong classifiers, while the response to an input pattern is given as the maximum response of all K strong classifiers. Thus, a window is classified as positive if a single strong classifier yields a positive score and negative only if all strong classifiers consider the window as negative. Also the runtime is only linear in the number of weak classifiers. During the learning phase positive samples which are misclassified by all strong classifiers obtain a high weight, while positive samples which are classified correctly by a single strong classifier are assigned a low weight. This enables the learning algorithm to focus on a subpart of misclassified data (up to the current round) with a single strong classifier. Other strong classifiers are not affected and therefore do not loose their discriminative power on their specific clusters learned.

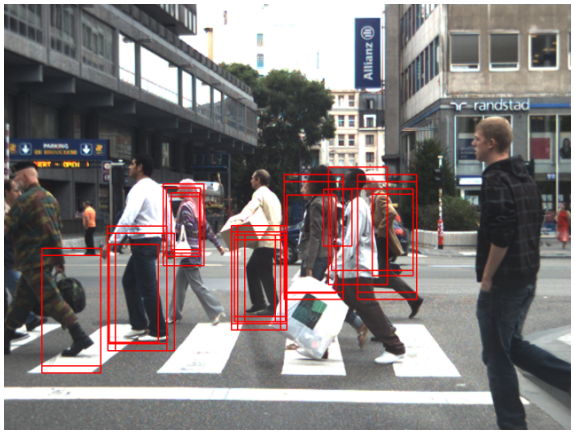


Figure 3.3: False positive detections with high scores before the bootstrapping stage. Detections close to pedestrians are true positives and not shown here.

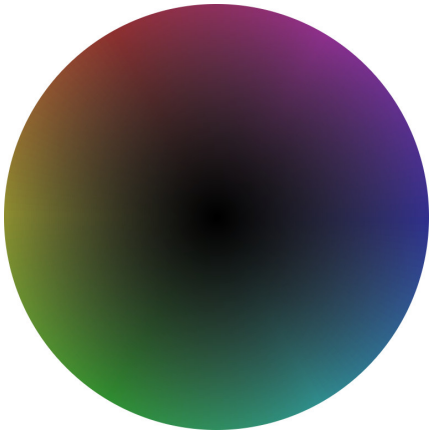


Figure 3.4: Color coding for optical flow. For a sample flow image see figure 3.6.

3.3 LEARNING AND TESTING

While features and classifiers are the key components of the detectors several issues need to be taken care of for both learning and testing. Those details are often crucial to obtain best performance, even though they are seldom discussed in literature. The following sections give some detailed insights on our learning (section 3.3.1) and testing procedure (section 3.3.2).

3.3.1 Improved Learning Procedure

Our classifiers are trained in a two-step bootstrapping process. First an initial classifier is trained on all supplied positive images and randomly cropped negative samples. Next all negative images are scanned in order to detect hard examples which are then added to the negative set which is followed by a second round of training.

In order to improve the statistics of hard examples for the domain where pedestrians actually appear, the negative test set also contains frames from an onboard camera recorded in an urban area. Those are scanned for hard examples, but detections that are close to a pedestrian in *x-y-scale*-space are considered true positive. The minimal distance is chosen such that detections on body parts are allowed as hard examples, because we observed in accordance with [10] that the final detector tends to fire on body parts.

Often these types of false positives are not well represented in other detectors' training data. Figure 3.3 shows highest scoring false positive detections in the bootstrapping phase after removing the full detections, showing that body parts

are indeed hard examples for the initial detector and should thus be considered as negative samples in the second training round.

Additionally, we found that merging the false positive detections on the negative images by mean shift is beneficial in several ways. First, the variability of false positive detections for the second round of training can be increased and the space of negative samples is covered well, while keeping the memory requirements reasonable. Second, false positive regions with a larger number of false detections are not overcounted since they will only be contained once in the training set and thus have the same weight as regions on which the detectors only fires a few times. This is consistent with the fact that for real-world systems the optimal image-based performance is sought and all false detections should be treated equally.

3.3.2 Testing

As it is desirable for real-world applications to detect pedestrians as soon as possible we are aiming to detect pedestrians as small as possible. Empirically we found that given appropriate image quality upscaling the input image allows for a better performance gain with respect to small detections than shrinking the detection window (cf. figure 3.2). Therefore, we upscale the input image by a factor of two which allows to detect pedestrians as small as 48 pixels with a 64×128 pixel detection window (the window contains context in addition to the pedestrian). Sliding-window based detection systems usually fire multiple times on true pedestrians on nearby positions in scale and space. These detections need to be merged in order to allow for a per-image based evaluation such as false positive per image (FPPI) or precision and recall (PR). Here, we adopt an adapted bandwidth mean-shift based mode seeking strategy [9] to determine the position in *x-y-scale*-space, but determine the final detection's score to be the maximum of all scores within the mode. While others (e.g. [10]) have used the kernel density to form the final score, we found the maximum to provide more robust results. While most of the time the performance is comparable, in some cases choosing the kernel density leads to a significantly decreased performance in particular for the motion-enhanced detector (cf. figure 3.10f). Another important issue is the estimation of the kernel density – in a scale pyramid setting with a constant pixel stride for every scale, detections on larger scales are sparser. Thus, contrary to [10] when computing the kernel density we omit the kernel volume's scale adaption for the normalization factor.

3.4 NEW DATASET

The sequences of Ess *et al.* [25, 28] are popular publicly available video sequences for pedestrian detection recorded from a moving platform. While those are realistic for robotics scenarios, they are less realistic for automotive safety applications. This is mainly due to the relatively small ego-motion and the camera's field of view

Figure 3.5: Positive sample crops and flow fields of TUD-MotionPairs.

which is focusing on the near range. In order to show results for a more realistic and challenging automotive safety scenario in urban environments, we captured a new onboard dataset (TUD-Brussels) from a driving car. Dollár *et al.* [18] also introduced a new onboard dataset but evaluates static features only.

At the same time there is no dedicated training set containing temporal image pairs which has sufficient variability to train a discriminative detector based on motion features. Thus, we additionally recorded a new training dataset containing pairs of images to compute optical flow (TUD-MotionPairs). Both new datasets have been made publicly available.

3.4.1 New Training Set

Our new positive training set (TUD-MotionPairs) consists of 1092 image pairs with 1776 annotated pedestrians (resulting in 3552 positive samples with mirroring), recorded from a hand-held camera at a resolution of 720×576 pixels. The images are recorded in busy pedestrian zones. While there are camera turns in the training set, there is no significant egomotion because the recording person was standing most of the time. Some samples are shown in figure 3.5. Note that contrary to INRIA Person [11] our data base is not focused on upright standing pedestrians but also contains a significant amount of pedestrians from side views which are particularly relevant in applications due to the possibility of crossing the camera's own trajectory.

Our negative training set consists of 192 image pairs. 85 image pairs were recorded in an inner city district, using the same camera as was used for the positive dataset at a resolution of 720×576 pixels, while another 107 image pairs were recorded from a moving car. For finding body parts as hard samples as described in section 3.3.1 we use an additional set of 26 image pairs, recorded from a moving vehicle containing 183 pedestrian annotations. We use this training set for all

Figure 3.6: Optical flow as an additional information source. Since the camera is moving, because of motion parallax even nonmoving obstacles like the two pedestrians to the left or the trees to the right are visible.

experiments throughout this chapter.

3.4.2 Test Sets

The new TUD-Brussels dataset is recorded from a driving car in the inner city of Brussels. The set contains 508 image pairs (one pair per second and its successor of the original video) at a resolution of 640×480 with overall 1326 annotated pedestrians. The dataset is challenging due to the fact that pedestrians appear from multiple viewpoints and at very small scales. Additionally, many pedestrians are partially occluded (mostly by cars and other pedestrians) and the background is cluttered (e.g. poles, parking cars and buildings and people crowds) as typical for busy city districts. The use of motion information is complicated not only by the fact that the camera is moving, but also by the facts, that the speed is varying and the car is turning. Some sample views are given in figure 3.7.

Additionally we evaluate our detectors on the publicly available ETH-Person [28] dataset. In [28], Ess et al. presented three datasets of 640×480 pixel stereo images recorded in a pedestrian zone from a moving stroller. The camera is moving forward at a moderate speed with only minor rotation. To evaluate our detectors, we only use the image recorded from the left camera. The sets contain 999, 450 and 354 consecutive frames of the left camera and 5193, 2359 and 1828 annotations respectively. As our detector detected many pedestrians below the minimum annotation height in these sets, we complemented the sets with annotations for the smaller pedestrians. Thus, all pedestrians with a height of at least 48 pixels are considered for our evaluation.

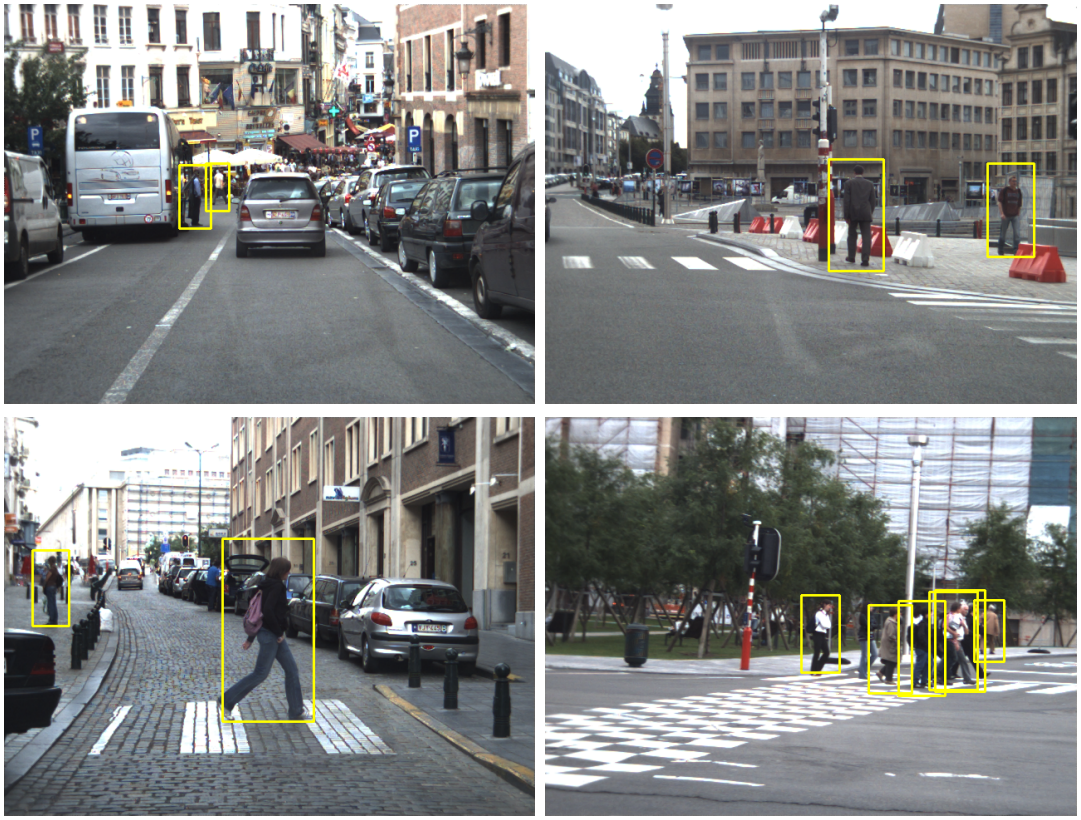


Figure 3.7: Detections obtained with our detector in an urban environment

3.5 RESULTS

Since we are interested in performance on a system level we refrain from evaluation in terms of FPPW but present plots in terms of recall and precision. This allows a better assessment of the detector as the entire detector pipeline is evaluated rather than the feature and classifier in isolation (cf. [18]). As a common reference point we will report the obtained recall at a precision of 90%. We also show plots of false positives per image to compare with previous work (i.e. [28]). We start the discussion of results with the static image descriptors and then discuss the benefit of adding motion features.

Results for the static features are given in the first row of figure 3.9 and figure 3.10. In combination with the HOG feature MPLBoost significantly outperforms AdaBoost on all tested sequences. In detail the improvement in recall at 90% precision is: 27.7 percentage points (pp) on ETH-01 (figure 3.9a), 24.4 pp on ETH-02 (figure 3.9d), 41.1 pp on ETH-03 (figure 3.10a) and 20.3 pp on TUD-BRUSSELS (figure 3.10d). Also it can be observed that HOG features in combination with MPLBoost do better than HOG features in combination with a linear SVM on all four datasets. The gain in detail in recall at 90% precision is: 8.5 pp on ETH-01 (figure 3.9a), 4.9 pp on ETH-02 (figure 3.9d), 22.6 pp on ETH-03 (figure 3.10a) and 2.0 pp on TUD-BRUSSELS

Figure 3.8: Sample detections for the different models learned by MPLBoost (K=4) using HOG, Haar, IMHwd. The models to the left respond more strongly to side/45-degree views, the models to the right to front/back views.

(figure 3.10d). Compared to a SVM with histogram intersection kernel (HIKSVM) the results are divergent. While HIKSVM outperforms MPLBoost by 1.4 pp on TUD-Brussels (figure 3.10d) and by 0.4 pp on ETH-01 (figure 3.9a), on ETH-02 and ETH-03 MPLBoost performs better by 1.9 pp(figure 3.9d) and 12.9 pp(figure 3.10a) respectively.

Next we turn to the results with HOG and Haar features in combination with different classifiers. On the TUD-Brussels dataset (figure 3.10d) we observe an improvement of 0.3 pp at 90% precision for MPLBoost, while on equal error rate (EER) the improvement is 4.3 pp. For the ETH databases we yield equal or slightly worse results compared to the detectors with HOG features only (figure 3.9a, (d), (a)). Closer inspection revealed minor image quality (cf. figure 3.12) with respect to colors and lighting on the ETH databases to be problematic, impeding a performance improvement (cf. figure3.9a, (d), (a)). Haar wavelets computed on color channels are not robust enough to these imaging conditions. Note however, that MPLBoost outperforms linear SVM, HIKSVM and AdaBoost for this feature combination showing its applicability for pedestrian detection. HIKSVM consistently obtained worse results with Haar features for static as well as for motion-enhanced detectors. Hence, these plots are omitted for better readability.

We continue to analyze the performance when IMHwd motion features in combination with HOG features are used for detection. The resulting plots are depicted in the second row of figure 3.9 and figure 3.10. For HIKSVM we observe a consistent improvement over the best static image detector. In detail the improvement at a precision of 90% precision is: 3.7 pp on ETH-01 (figure 3.9b), 16.9 pp on ETH-02 (figure 3.9e), 2.2 pp on ETH-03 (figure 3.10b) and 14.0 pp on TUD-Brussels (figure 3.10e). In contrast to [10] we can clearly show a significant performance gain on full images using motion features. The difference in performance however depends on the dataset and the distribution of viewpoints in the test sets. More specifically motion is beneficial mostly for side views but also for 45-degrees views whereas front-back views profit less from the added motion features.

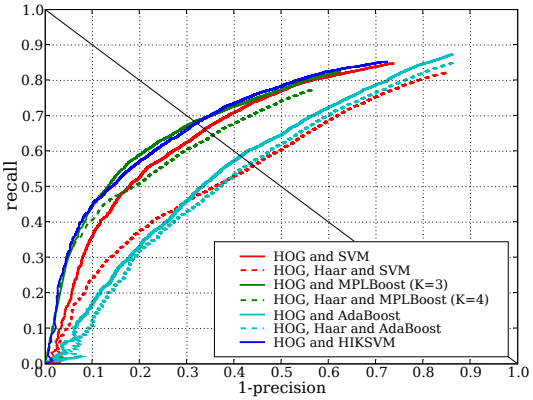
This explains the lower performance gain for ETH-01 (figure 3.9b) and ETH-03 (figure 3.10b) which are dominated by front-back views. We also observe that linear SVMs perform about as good as MPLBoost for this feature combination,

while HIKSVM does better than both except for ETH-03. Sample detections for MPLBoost and linear SVMs are shown in figure 3.11. Note that false detections differ between both classifiers. While MPLBoost tends to fire on high frequency background structure, SVMs tend to fire more often on pedestrian-like structures such as poles. We explain the similar overall performance by the fact that motion features allow a good linear separability of pedestrians against non-pedestrians in particular for side-views. This is consistent with our observation that MPLBoost mainly uses appearance features for the clusters firing on front-back views and more IMHwd features for clusters which fire on side views. Additionally, MPLBoost and SVMs again clearly outperform AdaBoost.

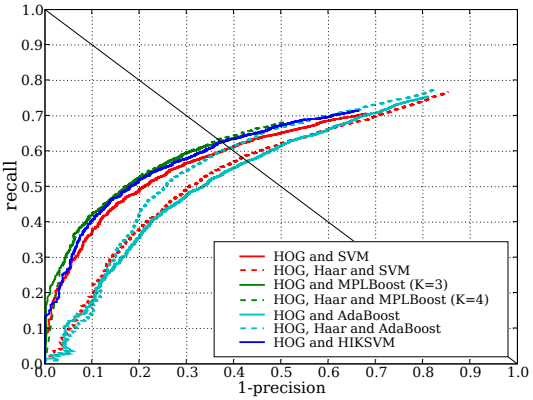
Combining IMHwd and HOG features additionally with Haar features yields similar results as for the static case with only little changes for MPLBoost. Interestingly linear SVMs obtain a better precision on TUD-Brussels for this combination, but loose performance on the ETH sequences as discussed for the static detectors. More sophisticated feature combination schemes (e.g. [44, 95]) may allow to improve performance more consistently based on multiple features, even if added features do not generalize well.

We have also analyzed the viewpoints different MPLBoost classifiers fire on. Figure 3.8 depicts high scoring detections on TUD-Brussels of the detector using HOG, IMHwd and Haar features for each of the four clusters. Two clusters predominantly fire on side and 45-degree side views while two clusters mostly detect pedestrians from front-back views.

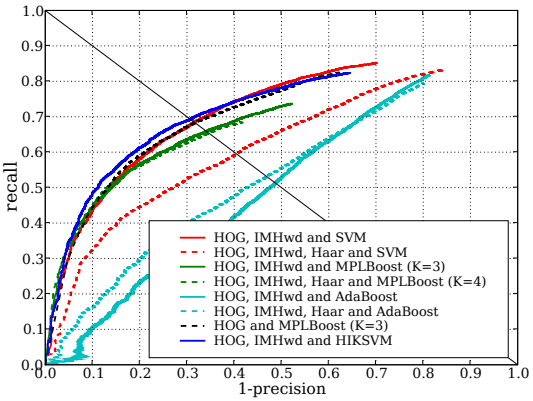
Finally, we compare our detector to the system of Ess et al. [28] (last row of figure 3.9 and figure 3.10). The original authors kindly provided us with their system's output in order to allow for a relatively fair comparison based on the modified set of annotations. For each sequence we plot the best performance of a static image feature detector and of the best detector including motion features. We consistently outperform Ess et al. [28] on all three sequences without any refinement of detections by the estimation of a ground plane. This refinement could obviously be added and would allow for further improvement. At 0.5 false positives per image we improve recall compared to their system by: 18.6 pp on ETH-01 (figure 3.9c), 32.2 pp on ETH-02 (figure 3.9f) and 37.3 pp on ETH-03 (figure 3.10c). To keep this comparison fair, we only considered pedestrians larger than 70 pixels similar to the original evaluation setting. Also note that HIKSVM with motion features clearly outperforms MPLBoost, while both classifiers are almost on par when all pedestrians as small as 48 pixels are considered. We also outperform Zhang et al. [122] who report 64.3% recall at 1.5 FFPI even though their detector is trained on ETH-02 and ETH-03 whereas our detector is trained on an independent and more general multi-view training set. Sample detections of our detector as well as system results of [28] are shown in figure 3.12. Note that our detector can detect very small pedestrians and achieves better recall throughout all scales by exploiting motion information.



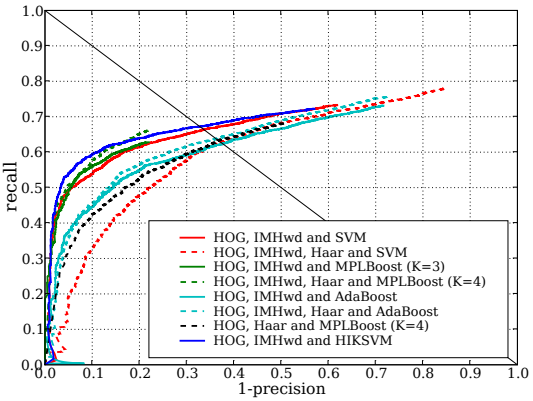
(a) Static image features (ETH-01)



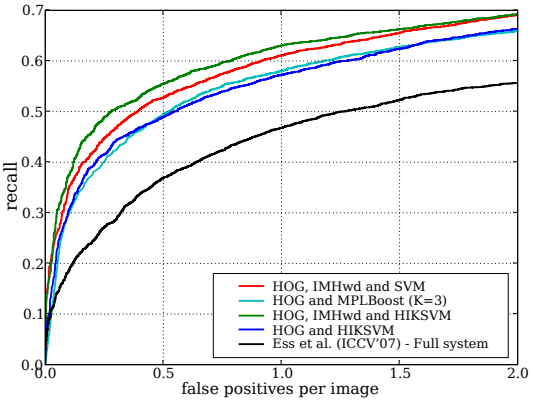
(d) Static image features (ETH-02)



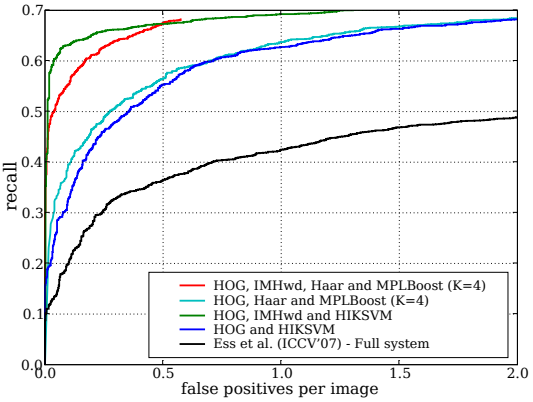
(b) Including motion features (ETH-01)



(e) Including motion features (ETH-02)

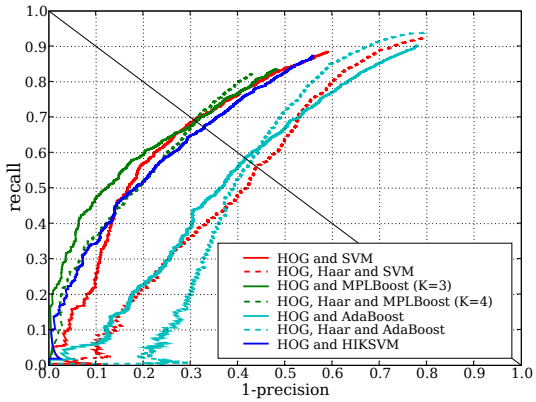


(c) Comparison to [28] (ETH-01)

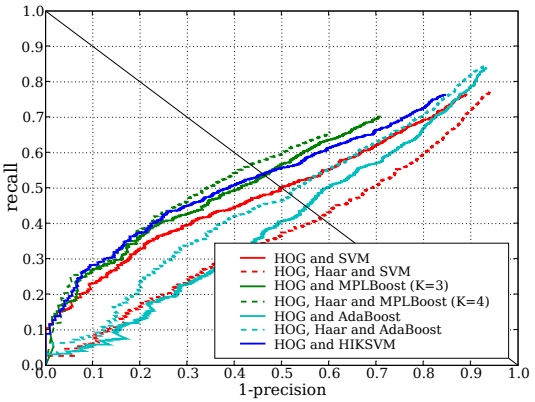


(f) Comparison to [28] (ETH-02)

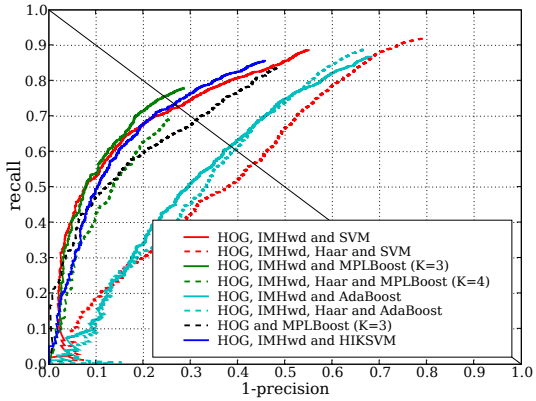
Figure 3.9: Results obtained with different combinations of features and classifiers on ETH-Person [28]. The first and second row show details on static and motion features in combination with different classifiers. Row three compares our detector to the system of [28].



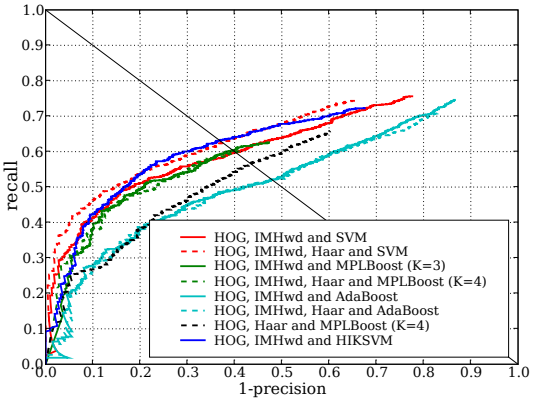
(a) Static image features (ETH-03)



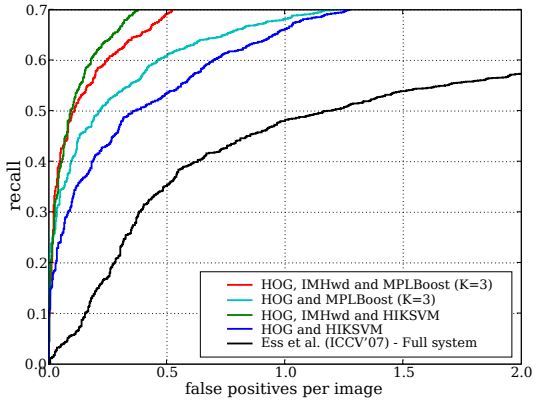
(d) Static image features (*TUD-Brussels*)



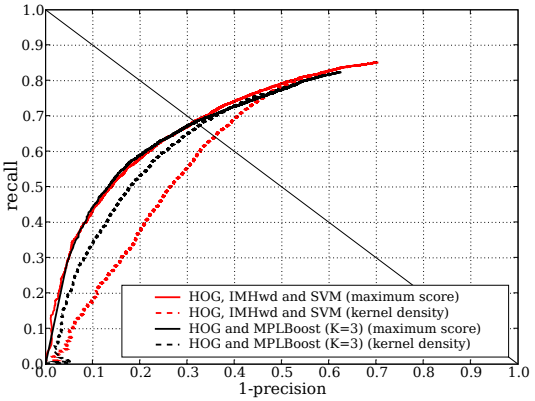
(b) Including motion features (ETH-03)



(e) Including motion features (*TUD-Brussels*)



(c) Comparison to [28] (ETH-03)



(f) Comparison of NMS scoring modes

Figure 3.10: Results obtained with different combinations of features and classifiers. The left column shows results on ETH-Person [28], the right column details the results on the new TUD-Brussels onboard dataset. The first and second row show details on static and motion features in combination with different classifiers. Row three compares our detector to the system of [28] and shows a comparison of different non-maximum suppression approaches (Fig. 3.10f).

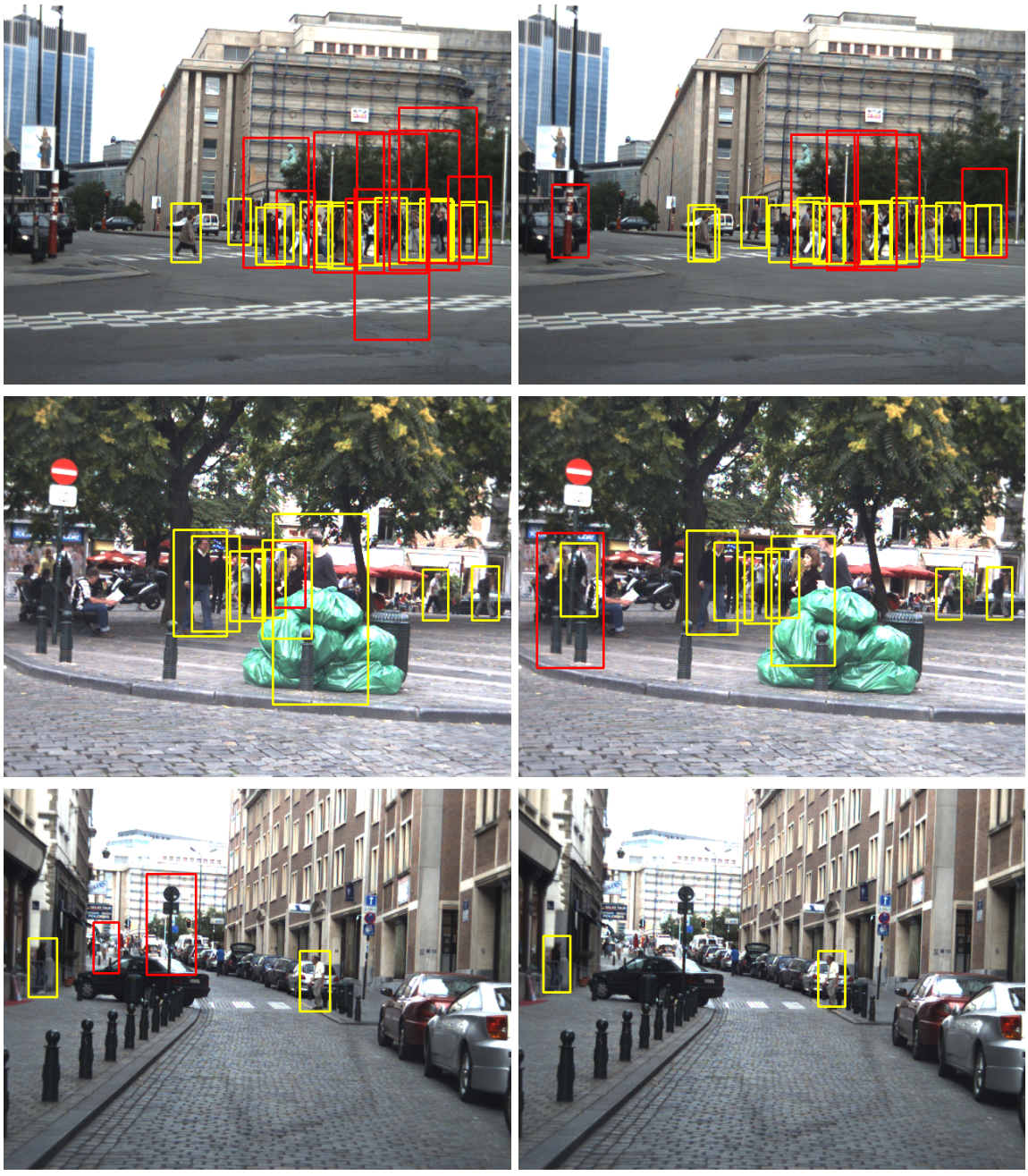


Figure 3.11: Sample detections on the TUD-Brussels onboard dataset at equal error rate for HOG, Haar, IMHwd and MPLBoost(K=4) (left column) and HOG, Haar, IMHwd and SVM (right column). True positives are yellow, false positives red.

Figure 3.12: Sample detections at 0.5 FPPI (First column: System of [28], Second column: Our motion-enhanced detector). Rows 1,2,3 correspond to figures 3.9c, 3.9f and 3.10c respectively, however all detections (even those smaller than 70 pixels) are shown. Note the false positive in the lower right image is actually a reflection of a true pedestrian.

3.6 CONCLUSION

In this chapter we tackled the challenging task of detecting pedestrians seen from multiple views from a moving car by using multiple appearance features as well as motion features. We showed that HIKSVM and MPLBoost achieve superior performance to linear SVM-based detectors for static multi-viewpoint pedestrian detection. Moreover, both significantly outperform AdaBoost on this task. When additional motion features are used, HIKSVMs perform best while MPLBoost performs as good as linear SVMs but in any case better than AdaBoost. In general however, MPLBoost seemed to be the most robust classifier with respect to challenging lighting conditions while being computationally less expensive than SVMs.

Dalal [10] found no improvement using the features of Dalal *et al.* [12] when employing them in a full-image detection setting. In contrast to that, our careful design of the learning and testing procedures improved detection performance on a per-image measure substantially when the IMHwd motion features of [12] are used. Two deficiencies in particular had to be addressed. First, the optical flow was computed on the whole frame during training instead of just being computed for a crop, which reduced boundary artifacts. Second, an error in the scale adaptation during non-maximum suppression was corrected, which manifested itself in particular when using motion features. The improvement is observed for pedestrians at all scales, but particularly for side views which are of high importance for automotive safety applications, since those pedestrians tend to cross the car's trajectory. Additionally, we show (contrary to [12]) that regularized flows [121], allow to improve detection performance. Adding additional Haar wavelets as features allowed to improve detection performance in some cases, but in general we observed that the feature is quite sensitive to varying cameras and lighting conditions.

4 NEW FEATURES AND INSIGHTS FOR PEDESTRIAN DETECTION

Contents

4.1 INTRODUCTION

The previous chapter focused on motion features and classifiers that are suitable for multi-view detection. While we also look at a new variant of the motion feature and its performance on degraded flow fields in this chapter, we focus on a new feature for static image detection, color self-similarity, and highlight important considerations for evaluation that, if ignored, can easily lead to wrong conclusions about relative detector performance.

The progress that has been made in detecting pedestrians is maybe best illustrated by the increasing difficulty of datasets used for development and benchmarking. The first [71] and second [11] generation of pedestrian databases are essentially saturated, and have been replaced by new more challenging datasets [18, 23, 112]. These recent efforts to record data of realistic complexity have also shown that there is still a gap between what is possible with pedestrian detectors and what would be required for many applications: in [18] the detection rate of the best methods is still $< 60\%$ for one false positive detection per image, even for fully visible people.

The present chapter makes three main contributions. First, we introduce a *new feature based on self-similarity of low-level features*, in particular color histograms from different sub-regions within the detector window. This feature, termed CSS, captures

pairwise statistics of spatially localized color distributions, thus being independent of the actual color of a specific example. The self-similarity allows to represent properties like "the color distributions on the left and right shoulder usually exhibit high similarity", independent of the actual color distribution, which may vary from person to person depending on their clothing. Adding CSS significantly improves state-of-the-art classification performance for both static images and image sequences. The new feature is particularly powerful for static images, and hence also valuable for applications such as content-based image retrieval. It also yields a consistent improvement on images sequences, in combination with optic flow.

The second main contribution is to establish a standard what pedestrian detection with a global descriptor can achieve at present, including a number of recent advances which we believe should be part of the "best practice", but have not yet been included in systematic evaluations. In evaluations on the two most challenging benchmarks currently available—CALTECH PEDESTRIANS [18] and TUD-BRUSSELS [112]—our detector achieved the best results at the time of publication of [103], outperforming published results by 5 to 20 percentage points.

Our third main contribution are two important insights that apply not only to pedestrian detection, but more generally to classifier-based object detection. The first insight is concerned with the fact that—for all classifiers—correct iterative bootstrapping is crucial. According to our experiments, the number of bootstrapping iterations is more important than the number of initial negative training samples, and too few iterations can even lead to incorrect conclusions about the performance of different feature sets. As a second insight, we point out some issues w.r.t. benchmarking and evaluation procedures, for which we found the existing standards to be insufficient.

4.2 DATASETS

For our evaluation, we focus on two databases, CALTECH PEDESTRIANS [18] and TUD-BRUSSELS [112], which are arguably among the most realistic and most challenging available datasets, CALTECH PEDESTRIANS also being by far the largest. INRIA PERSON is still a popular dataset, but it contains no motion, and consists mainly of large upright pedestrians with little occlusion.

The dataset CALTECH PEDESTRIANS contains a vast number of pedestrians—the training set consists of 192k (= 192000) pedestrian bounding boxes and the testing set of 155k bounding boxes, with 2300 unique pedestrians on 350k frames. Evaluation happens on every 30th frame. The dataset is difficult for several reasons. On the one hand it contains many small pedestrians and has realistic occlusion frequency. On the other hand the image quality is lacking, including blur as well as visible JPEG artifacts (blocks, ringing, quantization) which induce phantom gradients. These hurt the extraction of both gradient and flow features. For our evaluation we use the model trained on TUD-MOTIONPAIRS[112] (see below), and test on the CALTECH PEDESTRIANS training set. Some results for this setting—-train on external

data, test on the *Caltech* training set—have been published on the same website[1] as the database, and we got results for additional algorithms directly from Piotr Dollár for comparison. We will show that our enhanced detector using HOG, motion, and CSS outperforms all previously evaluated algorithms by a large margin, often by 10 percentage points or more.

The other test set, TUD-Brussels, contains 1326 annotated pedestrians in 508 image pairs of 640 × 480 pixels recorded from a car moving through an inner city district. It contains pedestrians on various scales and from various viewpoints. It comes with a training set (TUD-MotionPairs) of 1776 annotated pedestrians seen from multiple viewpoints taken from a handheld camera in a pedestrian zone, with a negative dataset of 192 images partially taken from the same camera and partially from a moving car. This training set is used for all experiments except for those on INRIAPerson (where the corresponding training set is used).

4.3 METHODS

As mentioned above, both feature and classifier choice strongly influence the performance of any sliding-window based method. In the following we describe the employed features including our proposed new feature based on self-similarity as well as our modifications of the histograms of flow (HOF) feature. This section also describes the classifiers and the training procedure used in the evaluation.

4.3.1 Features

Obviously, the choice of features is the most critical decision when designing a detector, and finding good features is still largely an empirical process with few theoretical guidelines. We evaluate different combinations of features, and introduce a new feature based on the similarity of colors in different regions of the detector window, which significantly raises detection performance. The pedestrian region in our detection window is of size 48×96 pixels. As it has been shown to be beneficial to include some context around the person [11] the window itself is larger (64×128 pixels).

HOG

Histograms of oriented gradients are a popular feature for object detection, first proposed in [11]. They collect gradient information in local cells into histograms using trilinear interpolation, and normalize overlapping blocks composed of neighbouring cells. Interpolation, local normalization and histogram binning make the representation robust to changes in lighting conditions and small variations in pose. HOG was

[1] http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/

recently enriched by Local Binary Patterns (LBP), showing a visible improvement over standard HOG on the *INRIA Person* data set [105]. However, while we were able to reproduce their good results on *INRIA Person*, we could not gain anything with LBPs on other datasets. They seem to be affected when imaging conditions change (in our case, we suspect demosaicing artifacts to be the issue), see Fig. 4.2a and 4.2b. Hence, we have not included HOG-LBP in further evaluations. In our experiments we compute histograms with 9 bins on cells of 8×8 pixels. Blocksize is 2×2 cells overlapping by one cellsize.

HOF

Histograms of flow (HOF) were initially also proposed by Dalal et al. [12]. We have shown that using them (e.g. in [12]'s IMHwd scheme) complementary to HOG can give substantial improvements on realistic datasets with significant ego-motion. Here, we introduce a lower-dimensional variant of HOF, IMHd2, which encodes motion differences within 2×2 blocks with 4 histograms per block, while matching the performance of IMHwd (3×3 blocks with 9 histograms). Fig. 4.2d schematically illustrates the new coding scheme: the 4 squares display the encoding for one histogram each. For the first histogram, the optical flow corresponding to the pixel at the i^{th} row and j^{th} column of the upper left cell is subtracted from the one at the corresponding position of the lower left cell, and the resulting vector votes into a histogram as in the original HOF scheme. IMHd2 provides a dimensionality reduction of 44% (2520 instead of 4536 values per window), without changing performance significantly. We used the publicly available flow implementation of [107][2]. In this chapter we show that HOF continues to provide a substantial improvement even for flow fields computed on JPEG images with strong block artifacts (and hence degraded flow fields).

CSS

Several authors have reported improvements by combining multiple types of low-level features [16, 80, 112]. Still, it is largely unclear which cues should best be used in addition to the now established combination of gradients and optic flow. Intuitively, additional features should be complementary to the ones already used, capturing a different part of the image statistics. Color information is such a feature enjoying popularity in image classification [94] but is nevertheless rarely used in detection. Furthermore, second order image statistics, especially co-occurrence histograms, are gaining popularity, pushing feature spaces to extremely high dimensions [80, 106].

We propose to combine these two ideas and use second order statistics of colors as additional feature. Color by itself is of limited use, because colors vary across the entire spectrum both for people (respectively their clothing) and for the background,

[2] In the previous chapter we used the optic flow software of [121], which is a precursor of [107]. We used the updated flow library for purely technical reasons. In our experiments we did not experience significant differences in detection performance between the two.

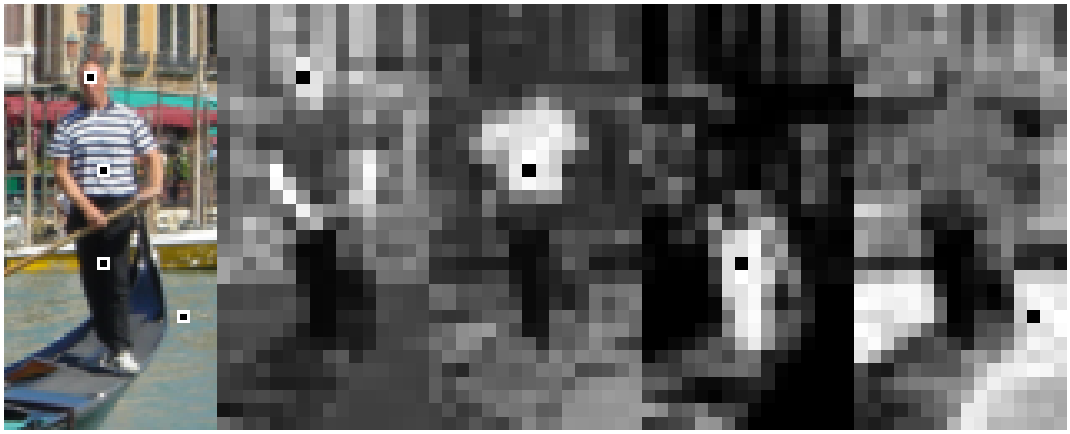
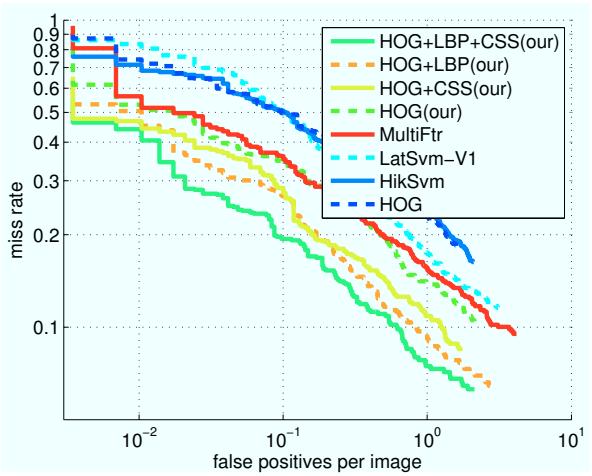


Figure 4.1: CSS computed at marked cell positions (*HSV*+histogram intersection). Cells with higher similarity are brighter. Note how self-similarity encodes relevant parts like clothing and visible skin regions.

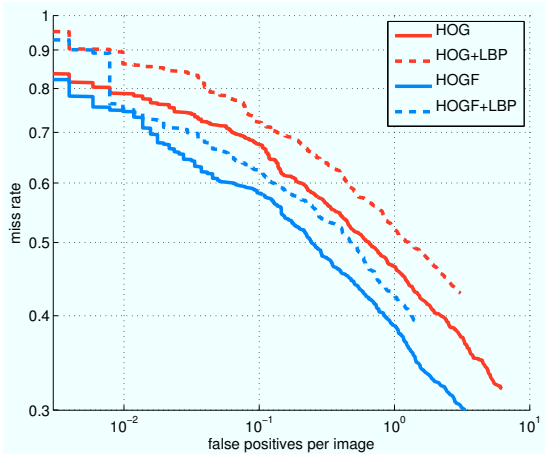
and because of the color constancy problem. However, people *do* exhibit some structure, in that colors are locally similar—for example (see Fig. 4.1) the skin color of a specific person is similar on their two arms and face, and the same is true for most people's clothing. Therefore, we encode color *self-similarities* within the descriptor window, i.e. similarities between colors in different sub-regions. To leverage the robustness of local histograms, we compute D local color histograms over 8×8 pixel blocks, using trilinear interpolation as in HOG to minimize aliasing. We experimented with different color spaces, including $3 \times 3 \times 3$ histograms in *RGB*, *HSV*, *HLS* and *CIE Luv* space, and 4×4 histograms in normalized *rg*, *HS* and *uv*, discarding the intensity and only keeping the chrominance. Among these, *HSV* worked best, and is used in the following.

The histograms form the base features between which pairwise similarities are computed. Again there are many possibilities to define similarity between histograms. We experimented with a number of well-known distance functions including the L_1-norm, L_2-norm, χ^2-distance, and histogram intersection. We use histogram intersection as it worked best. Finally, we apply L_2-normalization to the $(D \cdot (D-1)/2)$-dimensional vector of similarities. In our implementation with $D = 128$ blocks, CSS has 8128 dimensions. Normalization proved to be crucial in combination with SVM classifiers. Note that CSS circumvents the color-constancy problem by only comparing colors locally. In computation cost, CSS is on the same order of magnitude as HOF.

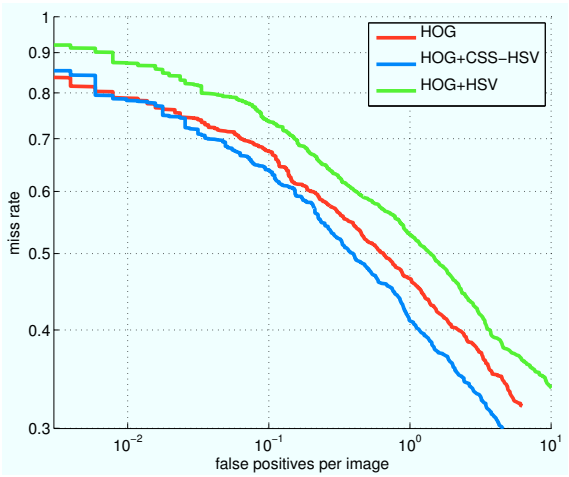
Fig. 4.2c supports our claim that self-similarity of colors is more appropriate than using the underlying color histograms directly as features. CSS in *HSV* space yields a noticeable improvement. On the contrary adding the color histogram values directly even hurts the performance of HOG. In an ideal world this behavior should not occur, since SVM training would discard un-informative features. Unfortunately this holds only if the feature statistics are identical in the training and test sets. In our setup—and in fact quite often in practice—this is not the case: the training data



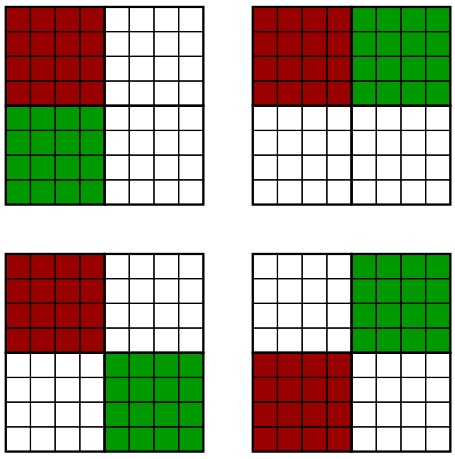
(a) LBP and CSS on INRIA PERSON



(b) LBP on TUD-BRUSSELS



(c) CSS vs. raw color (TUD-BRUSSELS)



(d) IMHd2 coding scheme

Figure 4.2: (a)-(c) Performance comparisons. Detections and labels on (a) are taken from the CALTECH PEDESTRIANS [18] website, plus ours. The classifier in (b,c) and for our curves in (a) is HIKSVM. (d) IMHd2 coding scheme for the pixelwise differences (4×4 cells are shown for simplicity, actual cell size is 8×8).

was recorded with a different camera and in different lighting conditions than the test data, so that the weights learned for color do not generalize from one to the other.

A similar observation was made in chapter 3, where we found that adding Haar features can sometimes help, but careful normalization is required, if the imaging conditions vary. Note that [16] do sucessfully utilize (raw) color, and in future work we plan to look into ways of incorporating it robustly into our detector (e.g. skin color may in principle be a sensible cue).

Note that self-similarity is not limited to color histograms and directly generalizes to arbitrary localized sub-features within the detector window. We experimented

with self-similarity on HOG blocks (see Fig. 4.3) as well as flow histograms, but we did not see significant gains.

4.3.2 Classifiers

We stick with those classifiers which performed best in recent evaluations [18, 112]: support vector machines with linear kernel and histogram intersection kernel (HIK), and MPLBoost [4]. Since AdaBoost did not yield competetive results, we chose not to include it here.

SVM

Linear SVMs remain a popular choice for people detection because of their good performance and speed. Non-linear kernels typically bring some improvement, but commonly the time required to classify an example is linear in the number of support vectors, which is intractable in practice. An exception is the (histogram) intersection kernel (HIK) [63], which can be computed exactly in logarithmic time, or approximately in constant time, while consistently outperforming the linear kernel.

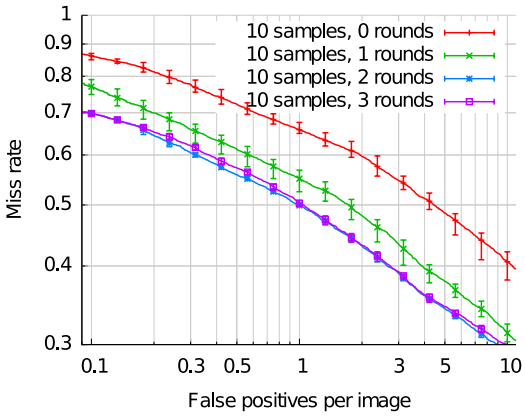
MPLBoost

Viola et al. [100] used AdaBoost in their work on pedestrian detection. However, it has since been shown that AdaBoost does not perform well on challenging datasets with multiple viewpoints [112]. MPLBoost remedies some of the problems by learning multiple (strong) classifiers in parallel. The final score is then the maximum score over all classifiers, allowing individual classifiers to focus on specific regions of the feature space without degrading the overall classification performance.

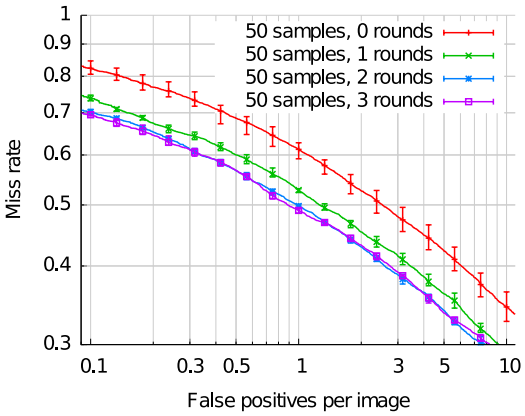
4.3.3 Training Procedure

A crucial point in training, which is often underestimated in literature, is the search for hard examples in the negative dataset, more specificially the number of retraining ("bootstrapping") iterations that are used. Dalal and Triggs [11] state that after one round "additional rounds of retraining make little difference so we do not use them". Felzenszwalb et al. [33] prove that repeated retraining leads to convergence for SVMs and repeat their training procedure—including the search for hard samples—10 times. Dollár et al. [16] use two bootstrapping rounds.

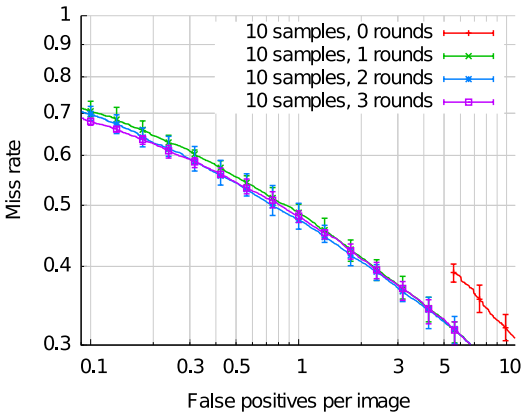
In Fig. 4.3a one can clearly see the influence of repeated retraining. Shown are the mean recall and maximum deviation for a fixed false positive rate, computed over five runs with different randomly selected sets of initial negative samples. The results are shown on TUD-Brussels for the HOG classifier paired with a linear SVM (chosen here because of its popularity). 10 negative samples are selected per



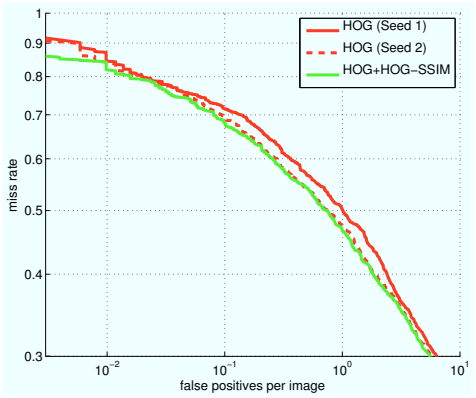
(a) 1920 initial negatives, linear SVM



(b) 9600 initial negatives, linear SVM



(c) 1920 initial negatives, MPLBoost



(d) Different random seeds, MPLBoost

Figure 4.3: Impact of additional retraining rounds.

training image at random, for a total of 1920 initial negative samples. Two results are immediately visible: with less than two bootstrapping rounds, performance depends heavily on the initial training set. In fact the variance is in the same order of magnitude as typical performance gaps between algorithms, leading comparisons *ad absurdum*. Furthermore, the figure shows that *at least two* retraining rounds are required to reach the full performance of the standard combination *HOG + linear SVM*.

One may argue that instead of additional bootstrapping rounds one could select more negative samples from the beginning. Fig. 4.3b shows that this is not the case: selecting 50 initial negatives per image (9600 total) somewhat alleviates the problem, but does not solve it. What's more, after 2—3 retraining rounds the advantages of using more initial samples vanishes, which confirms the strategy to concentrate on *hard* negatives.

For boosting classifiers (Fig. 4.3c)[3], the situation is worse: although mean perfor-

[3]The red curve is shortened because of precision issues – a lot of samples have scores very close

mance seems stable over bootstrapping rounds, the overall variance only decreases slowly—the initial selection of negative samples has a high influence on the final performance even after 3 bootstrapping rounds. Because of this, we show the superior performance of our new feature with HIKSVMs which have very good performance and where convergence during the iterative retraining phase is guaranteed [33]. We verified the required number of bootstrapping rounds experimentally.

Fig. 4.3d shows an example of a setting in which naive comparisons can even lead to unjustified conclusions. The used classifier is MPLBoost after 1 bootstrapping round, with either HOG features (red) or HOG plus self-similarity on HOG blocks (green). The only difference between the dashed and the solid green curve is the initial negative set. Had we only done one experiment with one bootstrapping round for this comparison, we might have come to the conclusion that self-similarity on HOG blocks helps significantly. It is important to make sure the result does not depend on the initial selection of negative samples, e.g. by retraining enough rounds with SVMs, as done in this chapter.

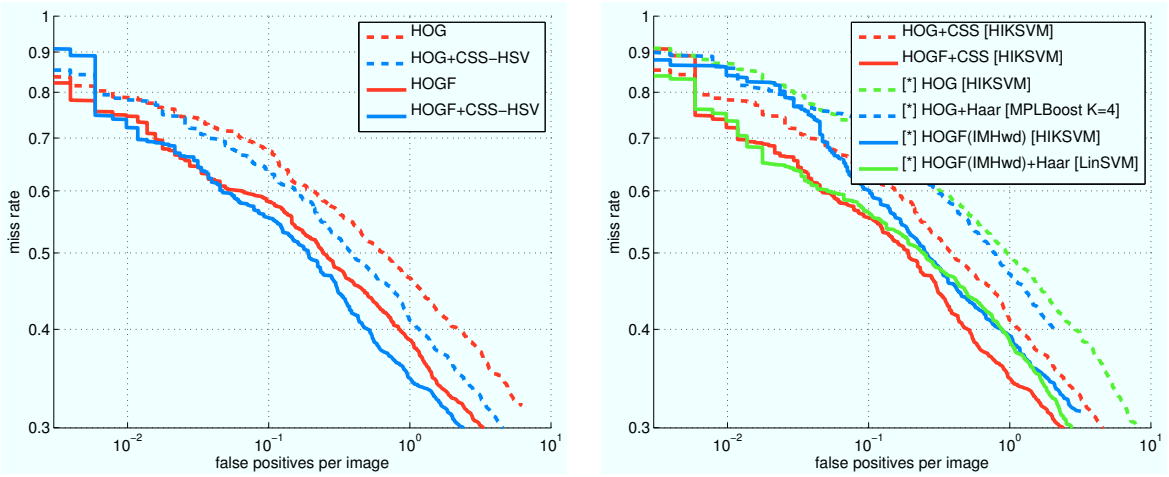
4.4 RESULTS

We continue with a detailed description of the results obtained with different variants of our detector. On CALTECH PEDESTRIANS, we used the evaluation script provided with the dataset. The plots (in Fig. 4.5) are stretched horizontally to improve readability (they end at 10fppi, instead of 100fppi as in the original publication). For TUD-BRUSSELS we evaluate on the full image, including pedestrians at the image borders (in contrast to [112]), who are particularly important for practical applications—e.g. for automotive safety, near people in the visual periphery are the most critical ones. Unless noted otherwise, the classifier used with our detector is HIKSVM.

Fig. 4.5e shows the performance on CALTECH PEDESTRIANS for the "reasonable" subset, which is the most popular portion of the data. It consists of pedestrians of $\geq$50 pixels in height, who are fully visible or less than 35% occluded. Our detector in its strongest incarnation, using HOG, HOF and CSS in a HIKSVM (HOGF+CSS), outperforms the previous top performers—the *channel features* (ChnFtrs) of [16] and the *latent SVM* (LatSvm-V2) of [33]—by a large margin: 10.9 percentage points (pp) at 0.01 fppi, 14.7 pp at 0.1 fppi and 7.0 pp at 1 fppi. Note that the interesting part of the plots is the left region, since more than 1fppi is usually not acceptable in any practical application.

We also point out that our baseline, HOG with HIKSVM, is on par with the state of the art [16, 33], which illustrates the effect of correct bootstrapping, and the importance of careful implementation. We did not tune our detector to the

to 1, which get mapped to 1 during serialization. This is not an issue for our detector since this vanishes after one round of bootstrapping and during training, no serialization happens (and all those examples are hard examples anyway).



(a) TUD-Brussels, impact of CSS

(b) Comparison with [112]

Figure 4.4: Evaluating on TUD-Brussels

dataset. Still, to make sure the performance gain is not dataset-specific, we have verified that our detector outperforms the original HOG implementation [11] also on INRIA Person (cf. Fig. 4.2a, also note that adding CSS provides an improvement over HOG+LBP).

HOG+CSS is consistently better than HOG alone, providing an improvement of 5.9 pp at 0.1fppi, which indicates that color self-similarity is indeed complementary to gradient information. HOG+HOF improves even more over HOG, especially for low false positive rates: at 0.1fppi the improvement is 10.9 pp. This confirms previous results on the power of motion as a detection cue. Finally, HOG+HOF+CSS is better than only HOG+HOF, showing that CSS also contains information complementary to the flow, and achieves our best result of 44.35% recall at 0.1fppi.

In Fig. 4.5f, the performance on the "near" subset (80 pixels or taller) is shown. Again, our baseline (HOG(our)) is at least on par with the state of the art [16, 33]. HOG+CSS provides better performance between 0.01 and 0.5 fppi, 6 pp at 0.1 fppi. Adding HOF to HOG (HOGF) adds 19.9 pp recall at 0.01 fppi. At 0.1fppi it beats the closest competitor HOG+CSS by 11 pp and the best previously published result [4] (LatSvm-V2) by 21.2 pp. Adding CSS brings another small improvement for large pedestrians. The reason that HOF works so well on the "near" scale is probably that during multi-scale flow estimation compression artifacts are less visible at higher pyramid levels, so that the flow field is more accurate for larger people.

Fig. 4.5c and 4.5d show the evaluation for increasing occlusion levels. Not shown are the plots for the "no occlusion" subset, which are almost identical to Fig. 4.5e, because only ≈5% of the "reasonable" pedestrians are partially occluded. Plots are also stretched vertically to provide for better readability.

Evaluated on the partially occluded pedestrians alone (which is not a significant

[4] at the time where this research work was done

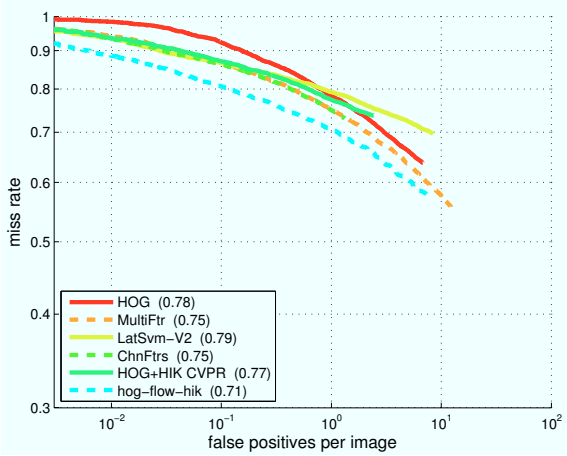
statistic, because there are only about 100 such examples, which is also visible in the jagged plots – every found pedestrian corresponds to an improvement of approx 1 pp), *latent SVM* and *channel features* slightly outperform our HOG, but again are dominated by HOG+HOF, with CSS again bringing a further small improvement.

On the heavily occluded pedestrians (Fig. 4.5d), the performance of all evaluated algorithms is abysmal. A lack of robustness to heavy occlusion is a well-known issue for global detectors. Still, the relative improvement with our detector is very noticeable: at 0.1 fppi, the recall of HOG+HOF+CSS is at 7.8% compared to 3.9% for *ChnFtrs*, doubling the recall. At 1fppi, our full detector still performs best, with 5.9 pp higher recall than LatSvm-V2. That color self-similarity helps in the presence of occlusion may seem counter-intuitive at first, because occlusion of a local sub-region is likely to affect its similarity to all other sub-regions. However, in the case of Caltech Pedestrians, "heavy occlusion" mostly means that the lower part of the body is occluded, so that similarities between different parts of the upper body can still be used.

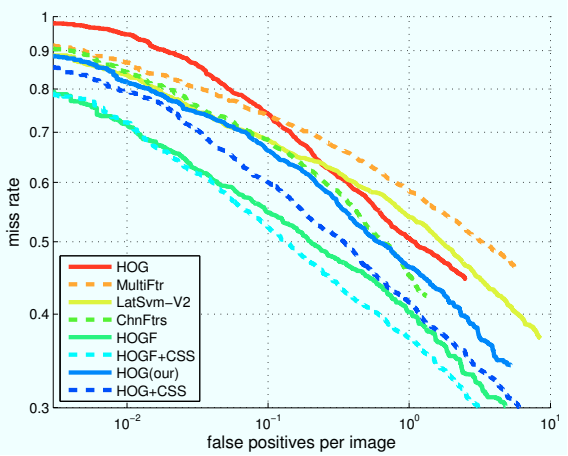
Fig. 4.4a shows the improvement gained by adding CSS on the TUD-Brussels dataset. CSS adds little in the high precision regime, but starting at 0.05 fppi there is a notable boost in performance, as recall is improved by 2.7 pp at 0.1fppi and 4.2 pp at 1 fppi. For static images with no flow information, the improvement starts earlier, reaching 3.6 pp at 0.1 fppi and 5.4 pp at 1 fppi.

Finally, Fig. 4.4b compares to the results of chapter 3 on TUD-Brussels. In this chapter Haar features did provide an improvement only on that dataset, on others they often cost performance. This is in contrast to CSS, which so far have produced consistent improvements, even on datasets with very different image quality and color statistics. Judging from the available research our feeling is that Haar features can potentially harm more than they help. We have nevertheless included the best results with *and* without using Haar features as reference.

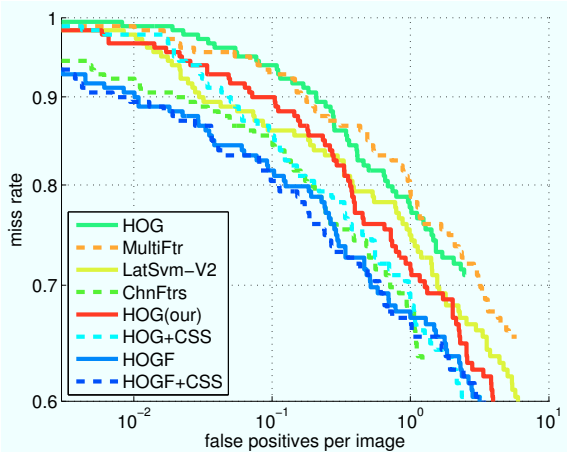
For the static image setting, HOG+CSS (dashed red) consistently outperforms the results of [112] by 5 pp–8 pp against HOG+Haar with MPLBoost (dashed blue), and by 7 pp–8 pp against HOG with HIKSVM (dashed green). Utilizing motion, the detector of [112] using HOG+HOF (in the IMHwd scheme), Haar features and a linear SVM (solid blue) is on par with HOG+HOF+CSS (solid red) for low false positive rates, but it starts to fall back at 0.2 fppi. The result of [112] using HOG+HOF with HIKSVM (solid green) is consistently worse by 3 pp–5 pp than HOG+HOF+CSS, especially at low false positive rates.



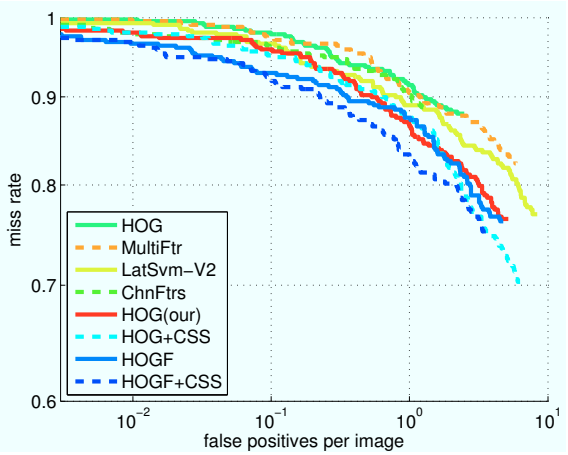
(a) Caltech-Pedestrians, "Overall"



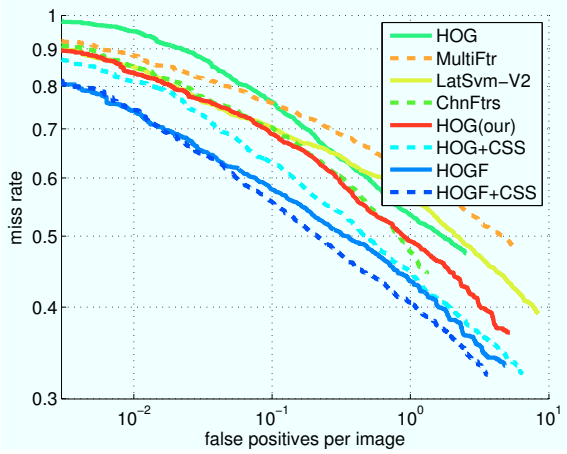
(b) Caltech-Pedestrians, "No Occlusion"



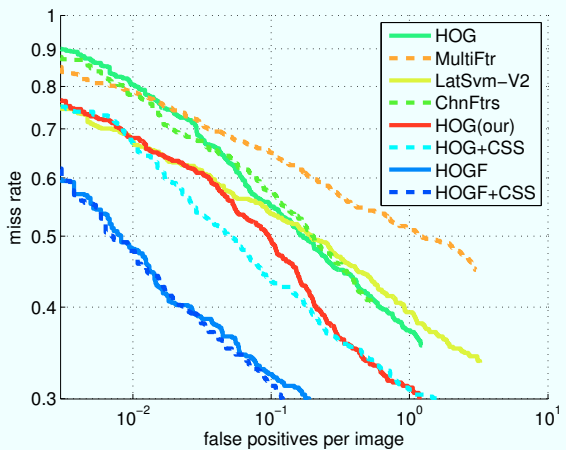
(c) Caltech-Pedestrians, "Partial Occlusion"



(d) Caltech-Pedestrians, "Heavy Occlusion"



(e) Caltech-Pedestrians, "Reasonable" subset



(f) Caltech-Pedestrians, "Near" subset

Figure 4.5: Evaluation under different occlusion conditions and size constraints on the Caltech Pedestrians dataset

4.5 SOME INSIGHTS ON EVALUATION

Another message of our investigation is that it is imperative to follow not only the same evaluation protocol, but to use *identical* scripts for the evaluation, in order to make results comparable, and even to make them meaningful at all. There are many design choices for evaluation scripts, some of which are often taken implicitely. Often, only the condition for two bounding boxes to match (e.g. the "PASCAL condition" [30], $\frac{\text{intersection}}{\text{union}} \geq 50\%$) is specified, which is not enough, as we will show.

We therefore suggest that the release of a dataset should always be accompanied by a suitable evaluation script, and that the raw detections should be published together with the corresponding curves. We have in all cases used the tools and detections used in the original publications [18, 112] for the respective datasets.

4.5.1 Evaluating on Subsets

Reliably finding every pedestrian in an image, regardless of size, is impossible even for a human. Therefore, and also to evaluate for a specific scale range or get rid of boundary effects, most of the time a subset of all annotated pedestrians is used in evaluations. This is often done in an underspecified way, and we will show how it distorts the results and introduces artifacts one has to be aware of.

As an example, let us examine the evaluation on CALTECH PEDESTRIANS using the "far" subset. In this setting, only pedestrians with an annotated height $20 \leq h < 30$ pixels are to be considered. Detections fulfilling the PASCAL condition can be as small as 10 pixels or as large as 59 pixels. Any annotation inside the 20–30 pixel range can be matched by a detection outside the range. This introduces an asymmetry which is difficult to handle. The CALTECH PEDESTRIANS evaluation script, as published at the same time as [18], discarded all detections outside the considered range, resulting in situations where a pedestrian with an annotated height of 29 pixels and a detected height of 30 pixels counts as a missed detection, although $\frac{I}{U} > 90\%$.

This is clearly undesirable, especially if many annotations are close to the size limit (which is always the case for small size ranges). However, trying to fix this bias introduces other ones. One possibility is to establish correspondence with the full sets of annotation and detection, and prune for size afterwards. For the discussion, we split the set of annotations and detections into *considered* and *ignored* sets based on the evaluation criteria. Annotations can fall into the *ignored* set because of size, position, occlusion level, aspect ratio or non-pedestrian label in the Caltech setting. Detections can fall into the *ignored* set because of size. E.g. if we wish to evaluate on 50-pixel-or-taller, unoccluded pedestrians, any annotation labeled as occluded and any annotation or detection <50 pixels falls in the *ignored* set.

The situation is relatively clear-cut for *considered* detections: if they match a *considered* annotation they count as true positive, if they match no annotation, or only

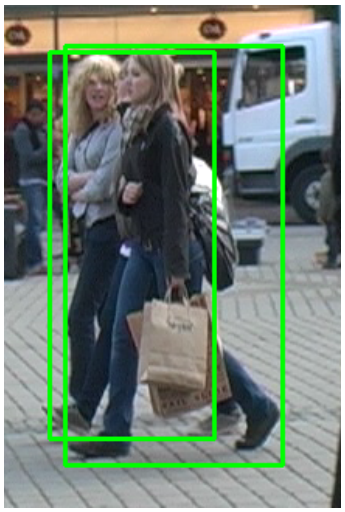


Figure 4.6: Pedestrians with overlapping bounding boxes so that $\frac{I}{U} \approx 0.59$.

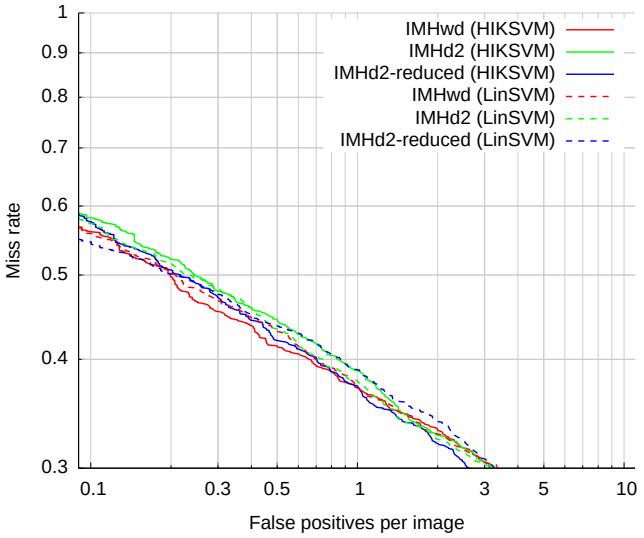


Figure 4.7: HOG&different variations of the HoF feature.

one that has already been matched to another detection[5], they count as false positive, and if they match an *ignored* annotation they are discarded. However, things are less clear for *ignored* detections: if an *ignored* detection matches an *ignored* annotation, it should be discarded. If an *ignored* detection matches no annotation, it seems reasonable to discard it, but this may introduce a bias, as will be seen shortly. If an *ignored* detection matches a *considered* annotation, applying the PASCAL condition suggests counting it as a true positive, and this is also the most consistent way to handle it over different settings (otherwise the same pedestrian could count as a false negative in the "far" setting, but as a true positive in the "overall" setting). However, allowing these matches introduces another problem: if one at the same time discards *ignored* detections matching no annotation, then the evaluation becomes vulnerable to (intended or unintended) exploitation: when, for example, one targets the "far" experiment, one could densely flood the image with detections just above/below the size limit. These will contain a valid match for every annotation inside the size range, but will be ignored if they do not match an annotation, leading to 100% recall without a single false positive. This effect is not limited to malicious flooding: parameter values that generate false detections on ignored scales will appear favourable, so iterative tuning could unintentionally introduce this bias[6].

The authors of [18] later refined the evaluation procedure in [19], introducing what they call "extended filtering", which expands the range of detections that get matched based on a parameter. For example, when evaluating on pedestrians of size

[5]The CALTECH PEDESTRIANS dataset has the concept of multi-*people* regions, which are allowed to match multiple detections.

[6]The evaluation script used in [112] is susceptible to this. However, since the detector did not output detections below the threshold, this has no effect on the results in [112].

50 and up, each detection 40 pixels and up (for $r = 1.2$) is counted as either a false or a true positive, and each detection below 40 pixels is discarded. This method tends to exhibit less bias, however the claim in [19] that this method is not exploitable is an overstatement. As each detector could chose not to evaluate below a certain size, if a detector returns a better result with a lower value of r it can obtain this by doing the filtering inside the detector system. Each system having a minimum in miss rate left from $r = 1.25$ in [19, figure 8] – which are most systems in this evaluation – could exploit this.

There is another issue worth noting: [18] try to match *considered* annotations preferably, even if an *ignored* annotation is a better match (higher overlap). This leads to artifacts when a pedestrian occludes another one so that their bounding boxes overlap sufficiently, as is the case in Fig. 4.6: If the occluding pedestrian is detected and the occluded pedestrian is not, the detection will match the unoccluded pedestrian in the "unoccluded" setting, but it will count as having detected the occluded pedestrian in the "occluded" setting. A more reasonable method would be to perform the matching without looking at the *ignored* attribute[7], aiming to optimize overlap, and doing the evaluation afterwards.

To summarize, there is no single correct way how to evaluate on a subset of annotations, and all choices have undesirable side effects. It is therefore imperative that published results are accompanied by detections, and that evaluation scripts are made public. As there are boundary effects in almost any setting (all realistic datasets have a minimum annotation size), it must be possible for others to verify that differences are not artifacts of the evaluation.

4.5.2 The Size Bias Introduced by the PASCAL Matching Criterion

There is also the possibility of a bias introduced by the matching criteria. For example, the mentioned PASCAL way – which is almost exclusively used at this time – of matching detections to annotations encourages overestimation of the object size. To illustrate this, it is worthwhile to look at the 1-dimensional case first, where y-position (0) and height (1) of the rectangle are fixed and known. Suppose we are aiming to find an object along the x axis, and have estimated its position and size (true size is 1). Figure 4.8 shows the value of $\frac{\text{intersection}}{\text{union}}$ over the distance d between true and estimated object center.

The shape of the curves is determined by the fact that the length of the intersection is given by the convolution of two rectangular impulses, which is trapezoidal. The matching measure is then

$$\frac{I}{U} = \frac{I}{\sum_i A_i - I} = \frac{I}{C - I}$$

where C is the sum of lengths (areas) of the intervals. Displayed are the three cases of correctly estimating the size of the object (blue), underestimating it (yellow) and

[7] For the Caltech setting, with the exception of *people* regions, which are to be matched separately

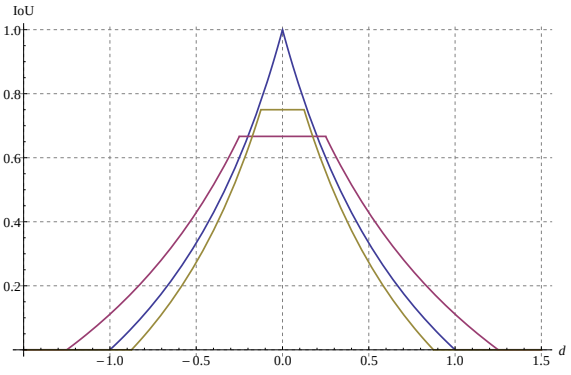


Figure 4.8: The PASCAL $\frac{I}{U}$ over distance for different detection sizes. Overestimating the size (purple) leads to a wider range of d where $\frac{I}{U} \geq \frac{1}{2}$ compared to correctly estimating the size (blue) or underestimating it (yellow).

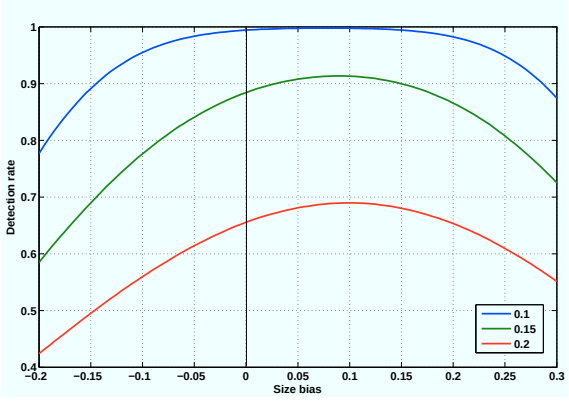


Figure 4.9: Detection rates over size bias for different amounts of localization noise. Adding a moderate size bias improves detection rates, especially if localization noise is high.

overestimating it (purple). For the case of the exactly matched size, the graph starts with value of 1 at $d = 0$, as the two rectangles are equal at this point. For $d = 0.25$, $\frac{I}{U}$ is 0.6. The threshold $\frac{I}{U} = 0.5$ is crossed at $d = \frac{1}{3}$. The yellow curve corresponds to underestimating the object width by 30%, so at $d = 0$ $\frac{I}{U}$ is 0.7. As the detected box moves inside the true box at the start, $\frac{I}{U}$ does not change for small values of d. However, when the borders of the boxes touch at $d = 0.15$, $\frac{I}{U}$ starts to drop, and is always smaller than the curve for the correct size. The purple curve corresponds to overestimating the size by 50%, so $\frac{I}{U}$ is $\frac{2}{3}$ for $d = 0$. For small d the true box is wholly contained in the detection, so $\frac{I}{U}$ does not change until $d = 0.25$ where the edges of the boxes touch. However, beginning at $d = 0.2$ we get a better $\frac{I}{U}$ than the blue curve, and the region where $\frac{I}{U} \geq \frac{1}{2}$ is significantly wider than in the case of the "correct" size. This means that a system producing *worse* results (because it consistently overestimates the size) can get *better* scores when measured using the PASCAL matching criterion.

One consequence of this is that it is possible to cheat in any setting where the PASCAL criterion is used. The impact of this cheating is dependant on the dataset and the detector – specifically on the uncertainties in position and scale. Figure 4.9 shows the detection rates over a size bias. Detector uncertainties[8] are modeled as independent gaussian noise (proportional to the size of the box) on the positions of the boxes' borders – scale variations are modeled implicitly (if the left border moves to the left and the right border moves to the right, the scale increases). The curves in blue, green and red correspond to standard deviations of 0.1, 0.15 and 2 respectively. As one can see, the bigger the uncertainty is, the more is to be gained

[8]Detector deficiencies are not the only source of localization noise – inaccurate annotations are also a factor.

from overestimating the size of the box, but increasing the size of the box by roughly 8% is a safe bet and increases the detection rate in all settings.

Another consequence is that if a system is trained truly to optimize the detection performance under the PASCAL criterion, it will aim – if its parameters allow it – to overestimate the bounding box size, which is obviously undesirable if one wants to train an accurate object detector.

What would be a better measure? Task-specific measures would be a possibility. For example, when locating in 3D with a calibrated camera, the position of the upper edge of the bounding box is not as crucial as the position of the lower one, because it is the latter that defines the position on the ground plane. One could implement a measure that reflects this, however the focus of this thesis is on general-purpose detectors. For object detection with a fixed aspect ratio, as done in this thesis, one possibility would be thresholding the distance in x-y-$\log s$-space[9], with the annotations' scale being defined as scale 1. This sacrifices the symmetry of the measure (which is okay since matching annotations to detections is an asymmetric task), but is not as prone to cheating. Since almost all related work uses the PASCAL measure, we will continue to use it to provide comparability, however one should keep this deficiency of it in mind when evaluating approaches.

4.6 CONCLUSION

This chapter advanced the state of the art in pedestrian detection in multiple ways: It introduced a powerful self-similarity feature to pedestrian detection, which – when applied to color channels – provides a noticeable improvement both in the single-frame setting and with additional motion information on two of the most realistic available pedestrian databases. A combination of carefully implemented HOG features, a variant of HOF to encode image motion, and the new CSS feature, together with HIKSVM as classifier, outperforms the state of the art published state of the art by 5–20 percentage points over a wide range of precision.

Concerning classifier training, we have shown that care has to be taken when comparing competing feature combinations: the improvement gained by introducing a feature can vary, and even vanish, as a function of a commonly underestimated parameter, the number of bootstrapping rounds.

On a meta-level we have pointed out that carefully specified evaluation procedures are needed in order to yield sensible performance metrics. Even seemingly harmless measures can introduce unwanted biases in the evaluation, and comparisons are essentially meaningless unless conducted with the same evaluation script, on raw detector outputs.

[9]The scale should be logarithmic because otherwise the distance to a box that is half as big would be as far as a box that is only 50% bigger.

5 DISPARITY STATISTICS FOR PEDESTRIAN DETECTION

Contents

5.1 INTRODUCTION

An important lesson from this thesis and previous research is that combining complementary cues is important to improve state-of-the-art performance. The previous chapters focused on the task of pedestrian detection in a setting where we have only one camera at our disposal. If a second camera is present, we can utilize the second view to obtain depth information, which is a very useful cue in object detection. How we use this information is the first major theme of this chapter.

Gavrila&Munder [42] and Ess et al. [25] combine appearance with stereo cues to detect pedestrians from moving vehicles, with the stereo components as modules for candidate generation and post-verification. In contrast to this, we directly incorporate stereo information into our detector.

We contribute a novel feature for pedestrian detection in stereo images, which we use in combination with the appearance and motion cues used in the previous chapters. Despite its simplicity, the new feature yields significant improvements in detection performance. It is also complementary to a good stereo feature previously studied in related work, which uses the HOG feature transform on the depth channel [75].

As the second focus of this chapter we explore the potential of classifier combination for pedestrian detection. While the combination of different features

[12, 25, 42, 112] has been key to much of the recent progress, the combination of different classifiers for the *same* feature has not been explored in the context of pedestrian detection to the best of our knowledge.

The benefit of both contributions is analyzed and discussed in detail using two different recent pedestrian datasets.

5.2 DATASETS

We use two different challenging datasets for our tests. Both databases have been recorded from a moving car in scenarios with many pedestrians: ETH-LOEWENPLATZ [25–27] and TUD-BRUSSELS [112]. Since we want to build a detector that utilizes both motion and stereo information, we are constrained in our choice of training data. We use two datasets: TUD-MOTIONPAIRS [112] and a new, auxiliary dataset to train the stereo-based component of our detector.

ETH-Loewenplatz. Our first test set consists of a video sequence of 800 consecutive stereo frames taken from a moving car, with annotations every 4 frames. In total it contains 2631 annotations, however we scan only for pedestrians bigger or equal to 48 pixels in size, which leaves us with 1431 annotations for evaluation.

TUD-Brussels. The second test set has 508 annotated frames recorded from a moving car. It originally had 1326 pedestrian annotations, but there were some small pedestrians missing. We supplemented those, resulting in a total of 1498 pedestrian annotations, with 1235 of them at least 48 pixels high. The dataset allows for optic flow estimation and has stereo image pairs.

TUD-MotionPairs. This dataset is used for training and contains 1776 pedestrian annotations in 1092 images, including the following frame for each annotated frame (to compute optical flow). The images are recorded in a pedestrian zone from a handheld camera, with pedestrians seen from multiple viewpoints. 192 image pairs without pedestrians, partly taken from a handheld camera and partly from a moving car, serve as negative set.

Auxiliary Training set. As TUD-MOTIONPAIRS does not contain stereo information, we have created a new dataset to train our stereo classifiers. The new dataset contains 2570 annotations in 824 frames in the positive set, with stereo and motion information available. However, most of the pedestrians in this set are small (2033 of them are smaller than the detection window, so that they have to be upscaled, resulting in suboptimal quality). The negative set contains 321 frames, again with motion and stereo information. The images have a resolution of 640x480 pixels and were recorded from a moving car (the same setup that was used for recording TUD-Brussels). Sample images are shown in Figure 5.1.

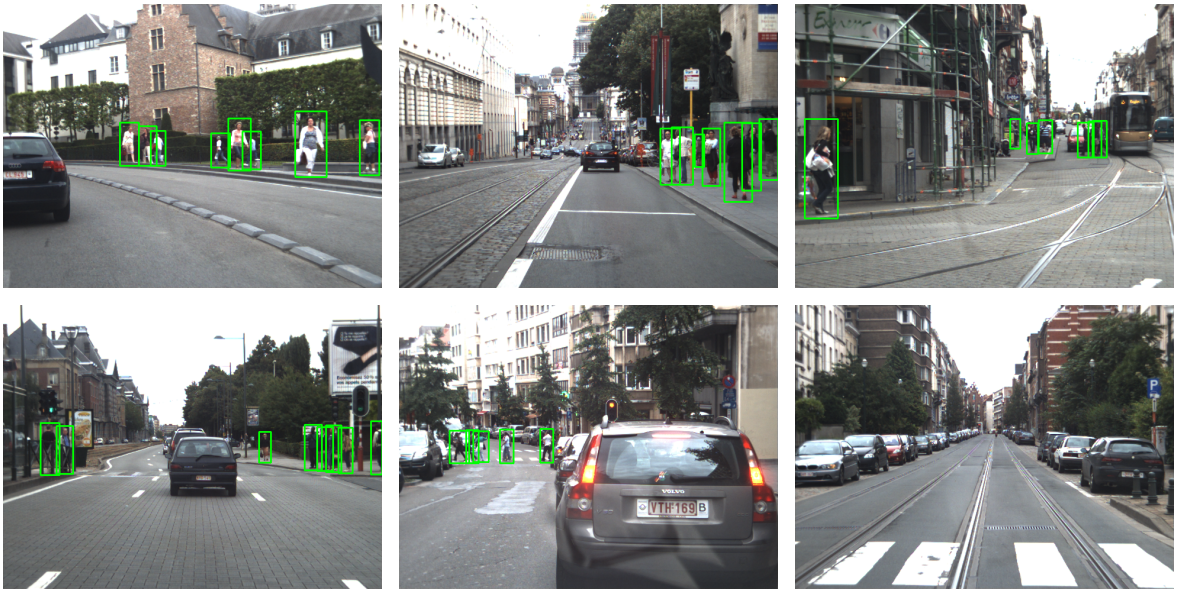


Figure 5.1: Sample images from the new auxiliary training set. The last image is from the negative set.

5.3 BASELINE FEATURES AND CLASSIFIERS

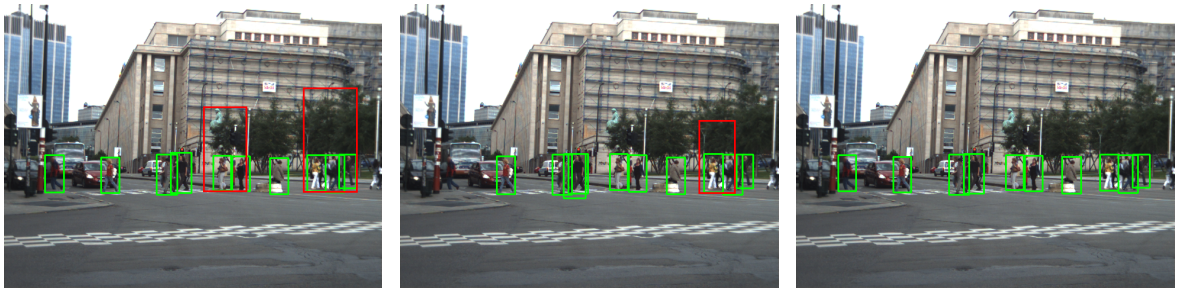
The set of features and classifiers we use as baselines includes HOG [11] and HOF [12] as features, and SVMs and MPLBoost [4, 54] as classifiers. The same features and classifiers were used recently in [112].

HOG. Dalal&Triggs proposed using histograms of oriented gradients in [11]. In HOG, every pixel votes for its gradient orientation into a grid of histograms using trilinear (spatial and orientation) interpolation. Local normalization is employed to make the feature robust against changes in illumination. Interpolation and histogramming makes the feature robust with regard to small changes in pose.

HOF. Histograms of Flow were introduced in [12] to encode motion information from optical flow. We use a reduced variant of the original IMHcd scheme with 2x2 blocks. Our version is on par with the original HOF in terms of performance. Flow fields are estimated with the publicly available optical flow implementation by Werlberger et al. [107].

SVM. Support Vector Machines are currently the standard for binary classification in computer vision. Linear SVMs learn a hyperplane that optimally separates negative and positive samples in high-dimensional feature space. Kernel SVMs are also possible, however their high computation time makes them intractable for sliding-window detection with high-dimensional feature vectors. An exception to this are histogram intersection kernels (HIKSVMs), for which an approximation can be evaluated in constant time [63].

MPLBoost. MPLBoost is an extension to AdaBoost[100], where K strong classifiers are learnt jointly, with each strong classifier focusing on a subset of the feature space.



(a) LinSVM (b) MPLBoost (c) LinSVM+MPLBoost

(d) LinSVM (e) MPLBoost (f) LinSVM+MPLBoost

Figure 5.2: MPLBoost and SVMs perform well but tend to have different false positives (a,b,d,e – red boxes correspond to false positives). By combining both classifiers the false positive rate can be reduced (c,f).

The final confidence is the maximum over the K classifiers, so only one of them needs to correctly identify a positive sample. Unless noted otherwise, we use $K = 4$ strong classifiers.

For training, negative samples are first randomly drawn from the negative training set to create an initial classifier. With this classifier the negative training images are scanned for *hard negatives* that get misclassified. These are added to the negative set and the classifier is retrained. We repeat this *bootstrapping* step twice to ensure that the result is minimally influenced by the random choice of the initial negative set.

The feature/classifier *components* we are using throughout this chapter were previously studied in chapter 3. Due to optimizations and changes in training procedure, there are some differences. Figure 5.8(b) compares the implementations. The three dotted lines compare the "old" HOG-detector (red dotted line) and HOG+Haar-detector (green dotted line) with our HOG-implementation (blue dotted line). Similarly the "new" HOG+HOF-feature (blue solid line) performs similar to or better than the previous HOG+HOF+Haar-feature (green solid line) and HOG+HOF-feature (red solid line). Note that we do not use Haar features as in chapter 3 we found them not to be beneficial in all cases.

5.4 COMBINATION OF CLASSIFIERS

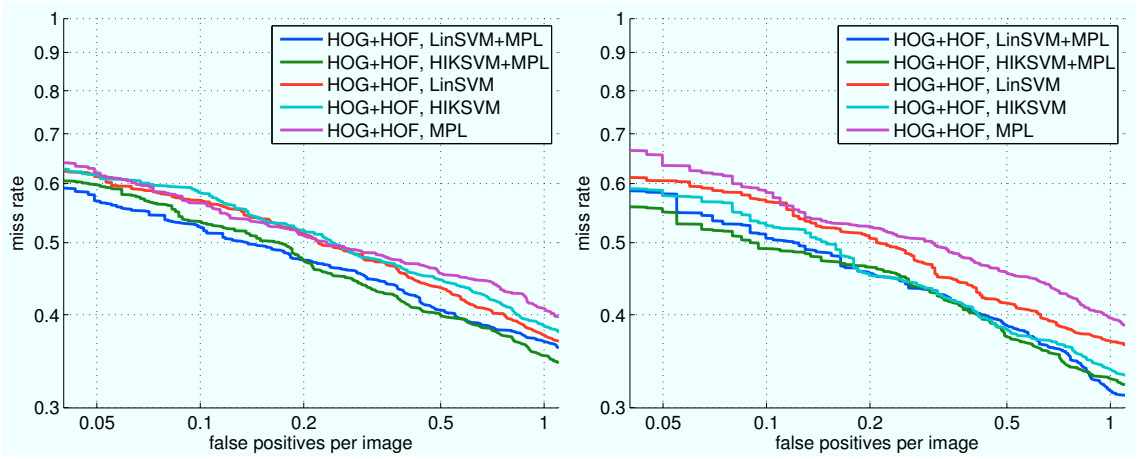
It is well-established that utilizing a combination of complementary cues significantly boosts detection performance. E.g. Gavrila et al. [42] use shape, texture, and stereo cues to build a detection system while Wojek et al. [112] use multiple features (including appearance and motion information) to boost detection performance. Rohrbach et al. [75] fuse classifiers separately trained on intensity and depth. In these cases, the complementarity of the classifiers results from the cues being from different sources (such as stereo and motion information) or from the sources being encoded into different features. However, those are not the only sources of complementary information.

In [112], we commented that MPLBoost and SVMs, while both giving good performance, tend to produce different false positives using the *same* feature set. For true positives, different classifiers are likely to give a positive answer, while for false positives the classifiers do not necessarily agree. See figure 5.2 for examples where LinSVM and MPLBoost (for the feature set HOG+HOF) produce different false positives (5.2(a,d) and (b,e) respectively). This gives a strong hint that by combining SVM and an MPLBoost classifiers, one can reduce the false positive rate. See figure 5.2(c,f) where such a combination eliminated false positives. This combination is described in the following.

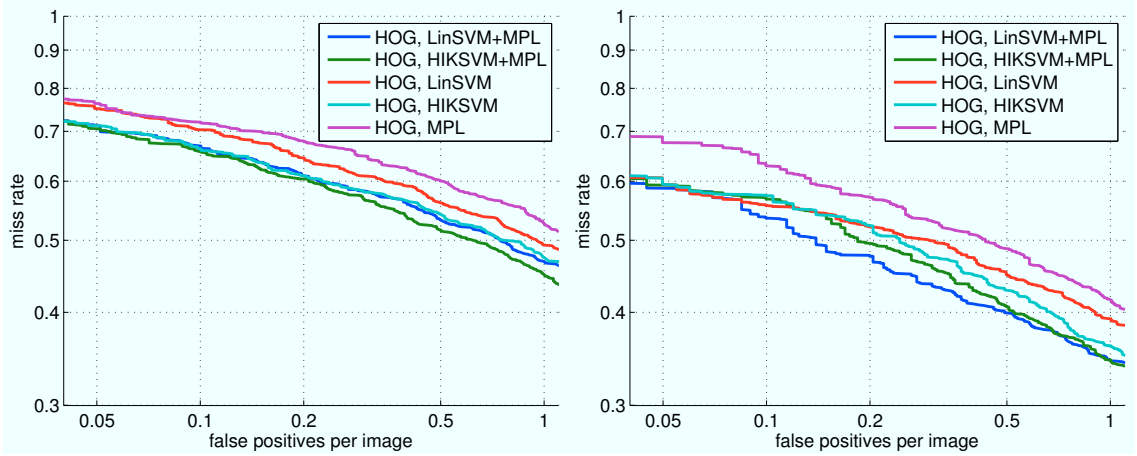
Starting from the above observation, this chapter explores the possibility to combine classifiers not only for different features but also to combine different classifiers for the same feature. Interestingly, in the context of digit recognition, Duin et al. [20] found that combining classifiers trained on the same features can be beneficial even though typically less beneficial than combining different features. The combination of classifiers for the same features is especially interesting as it is "cheap": Feature extraction is computationally expensive and often the bottleneck in today's systems. When combining classifiers on the same feature space, the feature vector has to be computed only once.

Classifiers are already combined at the training stage, which influences the bootstrapping phase: a window gets registered as a hard sample if it's hard for the *combined* classifier, enabling the classifiers to focus on data that is problematic for the final detector. This resulted in slightly better performance than training them separately. The combinations that we study in this section are linear SVM+MPLBoost and HIKSVM+MPLBoost, both trained on the same feature space, HOG+HOF. Combining a linear SVM with an HIKSVM did not show any improvement and thus is not reported here.

As noted before, one can expect classifier combination to improve classification if the combined classifiers have complementary characteristics. A (confidence-rated) classifier is a mapping from the feature vector space to a score. For an imperfect (but better than chance) classifier, the *probability density functions* (*pdf*s) of the positive and negative classes are overlapping. Under the reasonable assumption that the mean of the positive pdf is higher than the mean of the negative pdf, we can – without loss



(a) TUD-Brussels (b) ETH-Loewenplatz



(c) TUD-Brussels (d) ETH-Loewenplatz

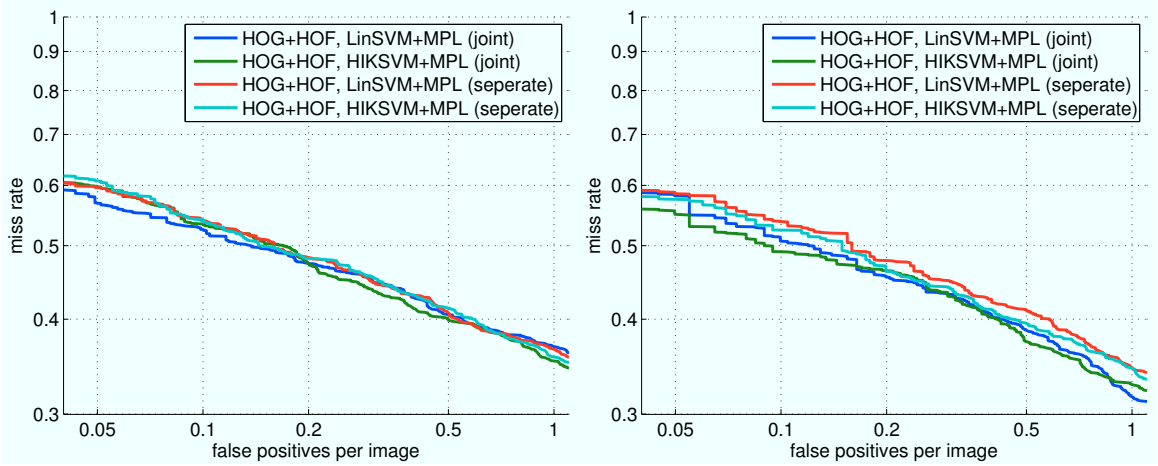
Figure 5.3: Results using classifier combination on TUD-Brussels and ETH-Loewenplatz with HOG+HOF and HOG alone as features. The single-component detectors are on par with the best published ones on TUD-Brussels from [112] (figure 5.8(b)), combining multiple classifiers yields a noticeable improvement.

of generality – rescale the mapping so that the means of the positive and negative pdfs are at +1 and -1, respectively. Classification errors (caused by the overlap of the pdfs) can then be expected to decrease when the variance decreases. The variance σ^2_{x+y} of a weighted sum $\alpha x + \beta y$ of classifiers x and y ($\alpha + \beta = 1$) for a given class is

$$\sigma^2_{x+y} = \alpha^2\sigma^2_x + 2\alpha\beta\sigma^2_{xy} + \beta^2\sigma^2_y$$

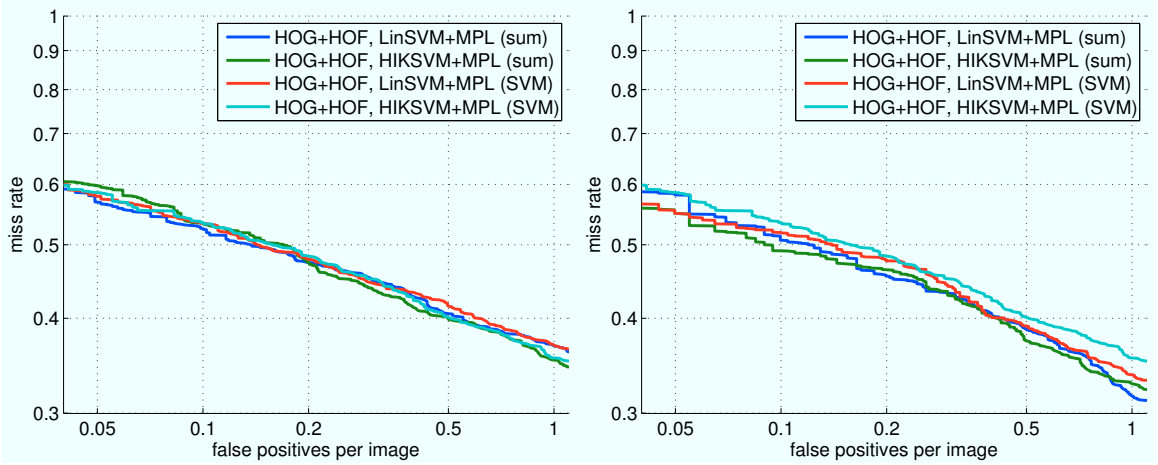
with σ^2_{xy} being the covariance. If this is lower than σ^2_x and σ^2_y, the combination can be expected to be beneficial.

Results are shown in figure 5.3 for the two test sets. For comparison, results for individual classifiers are shown as well. For TUD-Brussels and the feature



(a) TUD-Brussels (b) ETH-Loewenplatz

Figure 5.4: Training the combined classifier jointly is preferable to training them seperately.



(a) TUD-Brussels (b) ETH-Loewenplatz

Figure 5.5: Learning weights for the classifier does not improve upon simply adding the classifier scores.

combination HOG+HOF (Fig. 5.3(a)) the two combined classifiers (blue and green curves) clearly improve performance over the individual classifiers (red, cyan, violet curves). For ETH-Loewenplatz (Fig. 5.3(b)) the improvement of the combinations (blue, green curves) over the individual classifiers is also visible.

Note that sometimes linear SVM outperforms the intersection kernel SVM, which is counterintuitive. Remember however, that training and test set are recorded under different conditions such that learning on the training with a stronger classifier does not necessarily generalize better than a less powerful classifier.

At 0.1 false positives per image the best combined classifier for HOG+HOF (Linear

SVM + MPLBoost) has 4.2 percentage points (pp) more recall than the best single component classifier on TUD-Brussels, and 3.7 pp more recall on ETH-Loewenplatz. Using only HOG as feature, a smaller improvement can be observed over the best individual classifier for TUD-Brussels (see Fig. 5.3(c)) while on ETH-Loewenplatz the improvement is substantial at higher false positive rates: 5 pp improvement at 0.2 fppi (see Fig. 5.3(d)).

Training the component classifiers jointly as described above is beneficial. When the component classifiers are trained separately and combined for testing only (Fig. 5.4(a,b) red and cyan curves), performance is inferior to joint training (blue and green curves) especially for ETH-Loewenplatz (see Fig. 5.4(b)).

However, the improvement that can be gained depends on the employed features. If HOG features are used alone, the improvement is visible especially in the high recall/low precision region, as can be seen in Figure 5.3, while when HOG+HOF are employed as features, the improvement is mainly visible in the high precision region. Also, for HOG the improvement is greater on ETH-Loewenplatz than on TUD-Brussels.

The results reported so far have been obtained by averaging classifier scores as a confidence measure of the combined classifier. This gives both components equal weight. To see if performance improves when the weights are learned instead, we employ a linear SVM as a top-level classifier with the lower level classifier confidences as inputs. Here, 5-fold cross validation on the training set is used to train the top-level classifier without overfitting: we train on 80% of the training data and evaluate the component classifiers on the remaining 20%, with the cross-validation scores being the feature vectors for the top-level classifier. The final component classifiers are then trained using the whole training set. However, as can be seen in Figure 5.5, there is no significant improvement over equal weights, which is not surprising, as the classifiers work about equally well. As training takes significantly longer with this approach (≈ 6 times), we do not use it in the rest of the chapter. In the context of combining SVM kernels, [44] found that if the kernels are comparable in performance, averaging works well, while learning the combination is important when there are uninformative components, which agrees with our experience.

5.5 UTILIZING STEREO INFORMATION

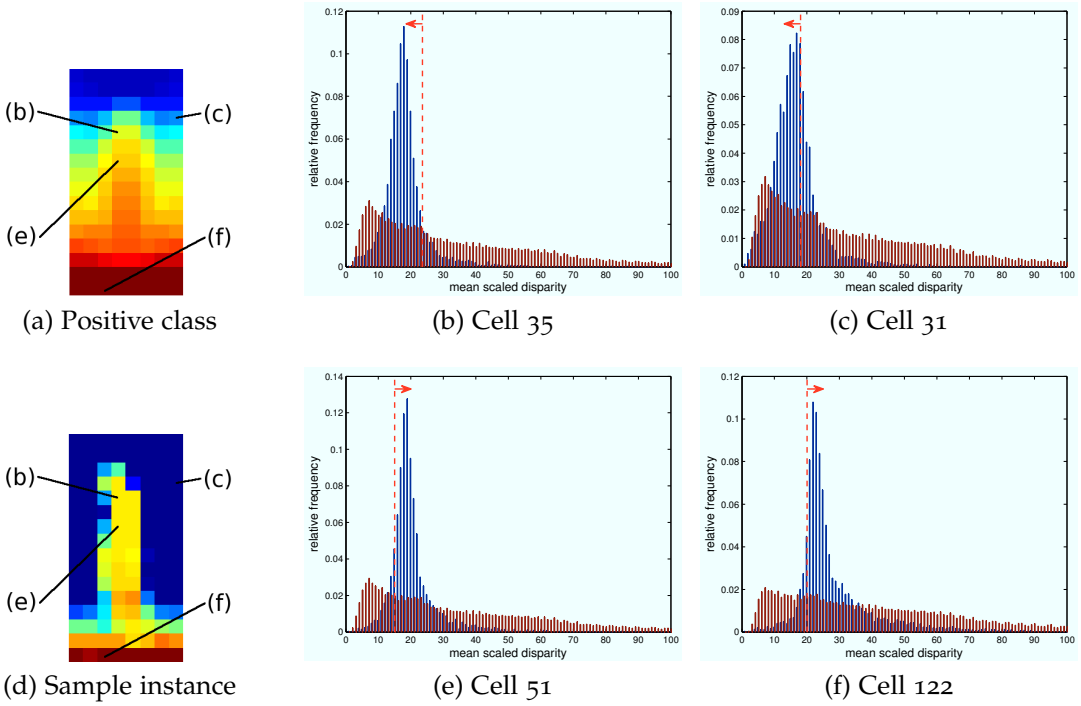
In the previous section, we showed that different classifiers on the same feature set can be combined to form a better classifier. However, the combination of different kinds of features from different sources of information promises a greater possible gain in information and consequently also in performance. One prominent source of information that is complementary to appearance and motion is binocular vision. Using a stereo image pair, we can extract disparity and depth information (see figure 5.6), which turns out to improve performance considerably.

Figure 5.6: Sample image with corresponding depth data. Near objects are blue, while far objects are dark red. Near the image borders, artifacts are visible.

5.5.1 HOS – HOG for Stereo

As a first stereo feature, we use a HOG/HOF-like feature. In [75], Rohrbach et al. computed the HOG descriptor on the depth field, which is inversely proportional to the disparity field, because its gradients are – in theory – invariant to the position of the pedestrian in the world. The gradients in the disparity image are not invariant (they are nonlinearly scaled). However, HOG is designed to provide invariance against scale changes in "intensity" (in this case, disparity). This becomes problematic only for very small disparities, where the nonlinearity are noticeable. On the other hand, using the depth also has its problems: since $Z \propto \frac{1}{d}$, small errors in disparity result in large errors of the depth map; moreover, pixels with disparity 0 have infinite depth and require special handling when building the descriptor, otherwise a single pixel can cause an infinite entry in the histogram. If we directly compute gradients on the disparity map, no special handling is required.

We have experimented with standard HOG descriptors (encoding small-range gradients in depth or disparity) and also with a variant of HOF on the disparity field, where we treat the disparity field like a vector field with the disparity as the x-coordinate and the y coordinate set to 0. The only relevant orientation bins here are the left and right bins: For every pixel, it is encoded if the pixels that are 8 pixels (in the L_∞ norm) away in horizontal, vertical or diagonal direction have a smaller or greater distance to the camera, weighted by the difference. This scheme in principle encodes less information than the full HOG descriptor, however stereo algorithms are not that accurate on a small scale, so long-range differences are more stable. Experimentally we did not observe any significant difference between the performance of this encoding and the encoding proposed by [75]. Therefore, in the following we use the HOF-like descriptor on the disparity field (termed HOS in the following) with a linear SVM as the classifier.



(a) Positive class (b) Cell 35 (c) Cell 31

(d) Sample instance (e) Cell 51 (f) Cell 122

Figure 5.7: Visualization of the Disparity Statistics feature. (a) is a color map of the median of the feature values over all positive samples (symmetric because training images get mirrored), (d) of an example training instance. Warmer color corresponds to bigger disparity/nearer points. Clearly, the feature is able to encode information like the pedestrian standing on the ground plane and the area around the upper body being more likely to be behind the pedestrian.

5.5.2 New Feature: Disparity Statistics

The disparity field has an interesting invariant property: in the pinhole camera model, the disparity d at a given point is

$$d = \frac{fB}{Z} \propto \frac{1}{Z} \tag{5.1}$$

with the focal length f, the baseline B, and the depth Z. The observed height h of an object of height H is

$$h = \frac{fH}{Z} \propto \frac{1}{Z} \tag{5.2}$$

Dividing equation 5.1 by 5.2, we get

$$\frac{d}{h} = \frac{B}{H} \tag{5.3}$$

This means that the ratio of disparity and observed height is inversely proportional to the 3D object height; for objects of fixed size that ratio is constant. The heights of pedestrians are not identical, but very similar for most pedestrians.

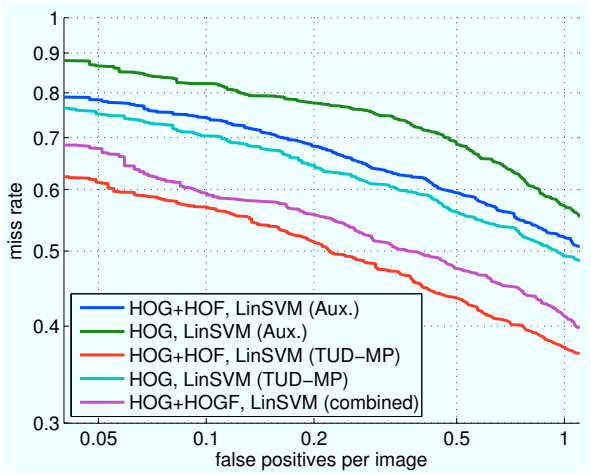


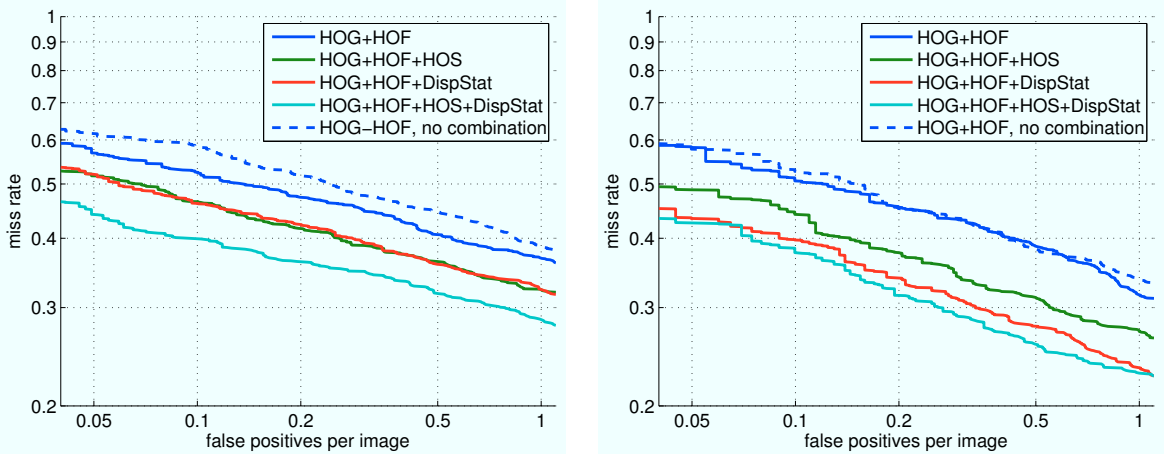
Figure 5.8: TUD-MotionPairs *(TUD-MP)* is a better training set than the auxiliary training set *(Aux.)* for appearance and motion information, however it contains no stereo information. Even combining TUD-MotionPairs with the auxiliary training set results in inferior performance for our detector when using appearance and motion as cues.

We can therefore, during sliding window search, divide the disparity values by the appropriate scale level determined by the layer of the image pyramid – e.g. for a reference height of 96 pixels and a scaled detection window of 64 pixels, disparities will be multiplied by 1.5. The scaled disparities of positive (pedestrian) samples will then follow a narrow distribution.[1]

This observation enables us to design a very simple and surprisingly effective feature. We divide the detection window into 8x8 pixel cells (the same as the HOG cell size, for computational efficiency). For each cell, the mean of the scaled disparities is computed. The concatenation of all 8×16 mean values from the 64×128 pixel window is the feature vector. For this feature, we use MPLBoost as classifier with $K = 2$ (more clusters did not help) and 100 boosting rounds.

Figure 5.7 visualizes the feature. In figure 5.7(a), the cell-wise median of all positive training samples is shown, 5.7(d) shows one particular positive training sample. One can immediately see different pieces of information captured by the new descriptor: the surrounding background is typically further away than the person, and the person usually stands on an approximately horizontal ground plane. In figure 5.7(b,c,e,f) statistics from example cells are shown along with weak classifier boundaries from the MPLBoost classifier. Displayed are the relative per-class frequencies of the disparity values. For the positive class, all 5140 training instances (including mirrored samples) are plotted, to plot the negative class 5 images were sampled densely, with the same parameters as in the sliding window search, resulting in 721900 samples (training of course uses all 321 images of the

[1] If the camera setup is different between the training and test images, the ratio between height and disparity has to be adapted accordingly to equations 5.1 and 5.2.



(a) TUD-Brussels (b) ETH-Loewenplatz

Figure 5.9: Results using stereo information on TUD-Brussels and ETH-Loewenplatz

negative set). The dashed red line shows the weak classifier threshold, with arrows to the right signaling a lower bound, and arrows to the left an upper bound. Note that they are *weak* classifiers – they are only required to work better than chance, so it does not matter if they miss-classify a portion of the training set. Even though the distributions overlap, making learning a non-trivial task, it is obvious that the class distributions are different and something can be learned from this data.

In figure 5.7(b) and (e), the disparity range for the upper body is evaluated by the weak classifiers, meaning the classifiers learn the size of a pedestrian (since the observed height is fixed – the height of the bounding box under evaluation – the *scaled* disparity relates inversely proportional to a height in 3D).

In 5.7(c), one weak classifier learned that the area to the right of the pedestrian usually is not closer to the camera than the pedestrian itself (note that the maximum of the distribution is at a lower disparity than the maxima of the distributions for (b) and (e)). However, the distribution here is not as narrow, because it is not uncommon that pedestrians stand next to other objects in a similar depth range. Figure 5.7(f) visualizes a weak classifier testing that the pedestrian stands on a ground plane, meaning that the cell under the pedestrian is closer to the camera than the pedestrian itself. Note that learning the pedestrian size and the ground plane assumption is completely data-driven.

5.5.3 Combining Classifiers for Different Cues

Finding a dataset to train a detector using depth, motion, and appearance is not trivial: The public designated training sets we are aware of don't have both stereo and motion information available. Our new training set, the *auxiliary training set*, has

this, however it is not as good as TUD-MotionPairs for appearance and motion, as can be seen in figure 5.8(a). The detector using HOG+HOF with a linear SVM has over 15 pp less recall when trained on this set (compare blue and red curves). Even joining the datasets for training results in inferior performance (violet curve).

To address this problem, we train different components on different datasets, and combine the components with an additional classifier stacked on top, which operates on the outputs of the components. In this section, we take the best combined classifier for appearance and motion (linear SVM + MPLBoost on HOG+HOF trained on TUD-MotionPairs) as one component. To combine the appearance/motion with the stereo components, a linear SVM is trained on top of the component outputs to provide the final score. The top-level SVM and the stereo-based classifiers are trained jointly using 5-fold cross validation on the auxiliary training set. To generate dense disparity maps, we used the algorithm of Zach et al. [120].

5.5.4 Results

As can be seen in figure 5.9, our new feature/classifier combination improves performance significantly. Best results from figure 5.3 are reproduced for reference: the dotted blue lines are the best performing individual classifier (HOG+HOF); the solid blue lines are the best performing combined classifier. On TUD-Brussels (Fig. 5.9(a)), the new disparity statistics feature combined with our HOG+HOF-classifier (red curve) performs as good as the HOS feature combined with HOG+HOF (green curve), resulting in an improvement of 6.4 pp recall at 0.1 fppi over the detector using HOG+HOF alone (blue curve). Combining both stereo features (cyan curve), the improvement is 12.6 pp over the HOG+HOF detector (solid blue curve), and more than 18 pp better than HOG+HOF with a linear SVM (dashed blue curve). The improvements are consistent over a wide range of false positive rates.

On ETH-Loewenplatz (Fig. 5.9(b)), adding HOS (green curve) results in an improvement of 6.6 pp at 0.1 fppi over HOG+HOF (blue curve). Using DispStat in addition to HOG+HOF (red curve) yields a higher improvement than HOS resulting in 11 pp improvement at 0.1 fppi. Further combining DispStat with HOS (cyan curve) in addition to HOG+HOF improves recall by another 2 pp. These results clearly show that DispStat is the stronger feature than HOS for this dataset. Compared to the best single-classifier detector with HOG+HOF as features (dashed blue), the overall improvement is 15 pp.

Comparing to state-of-the-art performance by [27] (they use a complete system integrating stereo, ground-plane estimation and tracking) our combined detector outperforms their best performance. In their evaluation scheme (pedestrians larger than 60 pixels) we outperform their system by about 5 pp at 0.1 fppi. This clearly underlines the power of the contributions of this chapter to improve the state-of-the-art in pedestrian detection.

In figure 5.10 sample results using stereo information are shown. In every pair, the upper image shows the HOG+HOF detector with HIKSVM+MPLBoost, the

Figure 5.10: Sample results using stereo information.

lower the full detector including HOS and the DispStat feature. Both detectors are shown at the point where they reach 70% recall, so differences are to be seen in the amount of false positives. The stereo features are especially good at eliminating false positives at the wrong scale, or not standing on the ground plane. "Typical" false positives, like car wheels (top left) and body parts (top right, bottom left) are easily filtered out, as well as detections having moving pedestrian as "legs" (bottom left). False positives on objects that are similar in 3d to a pedestrian are still an issue, for example the trash can with a traffic sign in the middle image in the lower row. Since the disparity field suffers from artifacts and missing information at the image border, some pedestrians (e.g. at the left border of the upper left image pair) are missed, however it detects others that the monocular detector misses (as both are tuned

to get 70% recall). Also note that in the lower left image the HOG+HOF detector overestimates the size of the pedestrian at the right image border, causing a false positive and a missed detection, while the detector using stereo features correctly estimates the size and position of the pedestrian.

5.6 CONCLUSION

This chapter consists of two contributions for pedestrian detection. First, we show that combining different classifiers trained on the same feature space can perform better than using a single classifier. Second, we introduce a new feature, called DispStat, for stereo, enabling the classifier to learn scene geometry information (like pedestrian height and the ground plane assumption) completely data-driven, without any prior knowledge. Combining those two contributions, we outperform the best published result on TUD-BRUSSELS by over 12 pp, in combination with an adaptation of HOG for disparity fields similar to [75], this increases to over 18 pp. We verified these results on a second challenging dataset, ETH-LOEWENPLATZ, where the performance of DispStat is even better, outperforming the HOS feature.

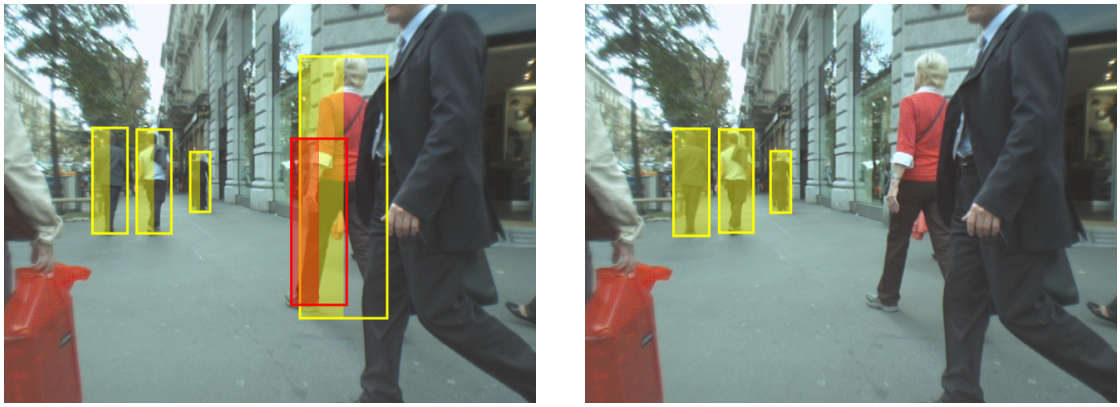
6 MONOCULAR SCENE UNDERSTANDING WITH EXPLICIT OCCLUSION REASONING

Contents

6.1 INTRODUCTION

The goal of this chapter is to enable reliable multi-object tracking from a moving platform in challenging real-world scenes even in cases when the objects are partially occluded for extended periods of time. Though by no means limiting the applicability, we focus on multi-people tracking, which is particularly challenging due to the large variability of human pose and appearance. The impressive progress in human detection and long-term tracking has allowed to detect and track several people simultaneously in complex scenes. Yet, state-of-the-art systems are still severely challenged by partial and full occlusions, which occur frequently in scenes of realistic complexity.

Typical multi-people tracking systems employ a Bayesian approach that relies on the robustness of both the human detection model and the tracking module. Without any explicit occlusion model such approaches have shown some robustness with regard to partial occlusions [5, 27, 42]. Elaborate association schemes have been proposed to enable recovery from partial and even full occlusions [29, 48, 53, 117]. However, these approaches are limited by the ability of their respective human detection model to detect and re-detect people before and after the occlusion, which limits their applicability to cases where people are sufficiently visible before and after the occlusion. In contrast, we explicitly address the problem of detecting and tracking people even when they are never fully visible or when they are significantly

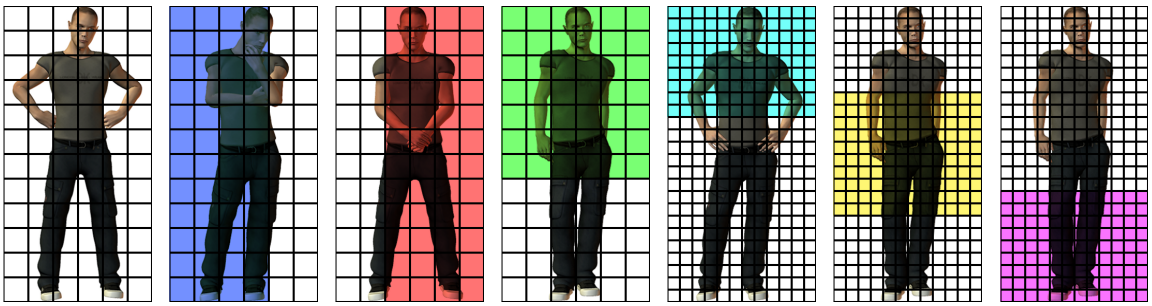


(a) Left-half detector (b) Full-body detector

Figure 6.1: Sample detections for models trained on the left half of a pedestrian and for the full-body detector. While the models for partial views do not perform as well overall, they are able to provide the scene model with hypotheses for partially occluded pedestrians.

occluded over long periods of time.

Drawing on successful prior work, we propose a new approach for multi-people tracking in the presence of challenging occlusions. The first important component is to track the complete scene rather than an assembly of individuals. This idea has been shown to enable robust 2D tracking of multiple people in surveillance scenarios [49, 86]. We adopt and extend this idea to *3D scene tracking using a monocular camera* [8, 109]. This is in contrast to other 3D tracking work [27] that uses stereo camera setups, yet is outperformed by our monocular system (see section 6.4). In order to enable detection and tracking of people even when they are never fully visible, we directly extend successful detection approaches such as HOG [11] and DPM [34] to enable the detection of partially visible humans for a variety of visibility scopes (see figure 6.2) and integrate them into our 3D scene model. This allows us to accumulate evidence not only for fully visible people but also for partially occluded people which can then be associated and tracked over extended periods of partial occlusion. Having a full 3D scene model allows us to determine the visibility of each individual in the scene, which in turn enables to predict which parts of the body are sufficiently visible and thus detectable. This allows us to define a novel complete 3D scene likelihood that tightly integrates full and partial human detectors within a 3D scene tracking framework. Quantitative experiments on publicly available data demonstrate that our model outperforms previous approaches and allows to associate and track people even in the presence of long-term partial occlusions.



(a) Full (b) Left (c) Right (d) Top (e) Top-HR (f) Mid-HR (g) Bot-HR

Figure 6.2: Detector regions.

6.2 DETECTORS

Our system uses seven detector components to provide the detection hypotheses. All components use HOG-like features [11], which have been proven to be a robust and effective feature for pedestrian detection.

The first detector component is the deformable parts model (DPM) by Felzenszwalb et al. [34]. It uses a combination of a global HOG template and several higher-resolution templates for parts that are allowed to vary in position relative to the position of the global template. This component performs best for fully visible large-scale pedestrians, but cannot handle small pedestrians and occlusions well. All other detectors are obtained by training an SVM on various parts of the HOG block grid of the detector window. They differ from [11] by an intersection kernel SVM instead of a linear SVM [63], using multiple rounds of retraining to make the training procedure stable [34] and an improved non-maximum suppression scheme [112] (those improvements were presented in chapters 3 and 4).

The SVM for three of the detectors is trained on the upper, left and right halves of the block grids (rounded up). The upper-body detector uses the top 8×7 blocks, and the left- and right-half body detectors the 15×4 left and right blocks, respectively (for an illustration see Fig. 6.2b–(d)). We also employ three models using a higher resolution detection window (256×128 pixels, resulting in a grid of 31×15 blocks), trained using only rows 3–12 (Top-HR), 11–20 (Mid-HR) and 20–29 (Bot-HR) (see Fig. 6.2e-(g)). These are motivated by the fact that in crowded scenes pedestrians are often quite close to the camera, but it is still desirable to detect them. All seven detectors are trained on the INRIA Person [11] dataset.

Note that the grid corresponds to a cell grid, and context cells are not shown. Rounding up when dividing the block grid in half means that, in the example of the left-half detector (Fig. 6.2b), one column of cells beyond the half is used (as the last column of blocks spans the 2 cell columns to the left and the right of the middle). However, since HOG operates on gradients/edges, and the edges from the right part of the body fall outside the used columns, the information used by this detector comes almost exclusively from the left side of the window.

It is important to note that training an SVM on parts, i.e. subsets of blocks, is different from using the model learned for the full body and only evaluating on the "visible" subset (which would be theoretically possible for additive kernels), especially because during the bootstrapping phase of training the detector finds hard samples for the partial-view models instead of hard samples for the full-body model. Even though a bank of detectors increases the computational load, we stress that the low-level feature representation can be shared among detectors and therefore only the classifiers need to be evaluated. To further reduce the load it may also be possible to adapt the DPM formulation to allow a tighter integration of our partial detectors; this will remain future work.

Fig. 6.1 shows detection examples for the left-half and full-body models. The full-body detector (Fig. 6.1b) is good at spotting fully visible pedestrians, but has problems finding pedestrians that are partially occluded. For these cases partial-view detectors can be beneficial, as they can spot partially occluded pedestrians (Fig. 6.1a). However, they typically also produce more false positives (as in Fig. 6.1a). As these tend to be inconsistent with the 3D scene model our method can discard them.

6.3 3D SCENE AND OCCLUSION MODEL

Before being able to introduce our explicit occlusion reasoning scheme based on 3D information, we first describe the basic 3D scene model. The 3D scene model is based on the work of Wojek *et al.* [109], which aimed to combine image evidence from detectors, geometric constraints and priors, as well as temporal reasoning to infer the 3D position of all objects in a scene from monocular video alone. For simplicity, we follow the notation of [109] and denote image coordinates in lower case, 3D world coordinates in upper case, and other vectors in bold.

For now assuming that only a single frame is given (the frame index is omitted for clarity), we take a Bayesian approach and define the posterior for the 3D scene $\mathbf{X}$ given image evidence $\mathcal{E}$ as

$$P(\mathbf{X}|\mathcal{E}) \propto P(\mathcal{E}|\mathbf{X}) \cdot P(\mathbf{X}) \tag{6.1}$$

where $P(\mathcal{E}|\mathbf{X})$ describes the observation model and $P(\mathbf{X})$ the prior assumptions about the 3D scene. The state of the 3D scene $\mathbf{X}$ is comprised of the individual objects $\mathbf{O}^i$, whose state is given by their 3D position $(O_x^i, O_y^i, O_z^i)^\top$ relative to the observer and their height H^i. The scene state $\mathbf{X}$ also includes the intrinsic and extrinsic camera parameters $\mathbf{K}$ and $\mathbf{R}$ (rotation only, see [109, Fig. 2]).

Prior.

We make the same basic assumptions as in [109], which apply to a variety of robotics and automotive scenarios: We assume that the camera is rigidly mounted to a platform, which along with all objects stands on a common ground plane ($O_z^i = 0$),

and has been calibrated off-line. The camera is furthermore assumed to undergo no roll and yaw w.r.t. the platform; odometer readings are used to determine the speed and turn rate of the platform itself. Hence the observer-centric coordinate system is fully specified by the pitch angle Θ, which may vary slightly as the platform accelerates or slows down.

Due to the low viewpoint in the sequences employed in this chapter, the correct estimation of distant objects is difficult requiring reliable estimates of the camera pitch. We also aim to avoid detecting background structures that stand on the ground but have incorrect height. To address these issues, we integrate prior knowledge. Specifically, the camera pitch Θ is assumed to follow a Gaussian distribution $\mathcal{N}(\Theta; \mu_\Theta, \sigma_\Theta)$ around the resting pitch μ_Θ. In addition, the height of each scene object H^i is assumed to follow a Gaussian distribution $\mathcal{N}(H^i; \mu_H^{c_i}, \sigma_H^{c_i})$, where $\mu_H^{c_i}$ denotes the mean height of the respective object class c_i. Consequently, the 3D scene prior can be written as

$$P(\mathbf{X}) \propto \mathcal{N}(\Theta; \mu_\Theta, \sigma_\Theta) \cdot \prod_i \mathcal{N}(H^i; \mu_H^{c_i}, \sigma_H^{c_i}). \tag{6.2}$$

Next, we turn to the observation model $P(\mathcal{E}|\mathbf{X})$. The image evidence $\mathcal{E}$ in this chapter is comprised of a set of candidate full object detections and a set of candidate object part detections. We will first describe our model for a single full body detector and then extend it to a setup with multiple part detectors. As we will see in the experiments, the combination of different detectors is beneficial for handling partial object-object occlusion as well as object truncation at the image boundary. Full object detectors return more reliable hypotheses than part detectors, but are limited to entirely visible objects. Frequently the detection confidence drops severely even when the object is only partially occluded or outside the image. Part detectors on the other hand allow to detect objects based on partial appearance, but also tend to produce a higher number of false positive detections.

Single detector likelihood.

In case of a single full body detector we define the likelihood following [109] as:

$$P(\mathcal{E}|\mathbf{X}) \propto \prod_i \Psi_D\left(\mathbf{d}^{a(i)}\right) \cdot \Psi_G\left(\mathbf{O}^i, \Theta; \mathbf{d}^{a(i)}\right) \tag{6.3}$$

Herein every 3D object hypothesis $\mathbf{O}^i$ is associated with one of the candidate detections $\mathbf{d}^{a(i)}$ via an association function $a(i)$. The appearance potential Ψ_D maps the detector's appearance score for the associated detection $\mathbf{d}^{a(i)}$ into the positive range. In practice we perform hard clipping of the SVM margin at zero (for negative scores). The potential Ψ_G models geometric constraints imposed by the ground plane, which is governed by the camera pitch Θ. In particular, a Gaussian in x-y-scale-space measures how well the projection of the object $\mathbf{O}^i$ to the ground plane, $\mathbf{o}^i$, matches the associated detection $\mathbf{d}^{a(i)}$:

$$\Psi_G(\mathbf{O}^i, \Theta; \mathbf{d}^{a(i)}) = \mathcal{N}(\mathbf{o}^i; \mathbf{d}^{a(i)}, \sigma_G + \bar{\sigma}_G). \tag{6.4}$$

The kernel's bandwidth is split into a constant component σ_G and a scale-dependent component $\bar{\sigma}_G$ to account for the sliding-window detector's discrete scanning stride.

6.3.1 Multi-Detector Likelihood with Occlusions

We now extend the above observation likelihood of [109] to include multiple detectors as evidence. To incorporate local part detections robustly we perform occlusion handling by explicitly leveraging 3D scene information. For each part p (we also refer to the full object detector as a *part* in the following) we compute its projection's expected visibility v_p^i based on the global 3D scene model. Assuming that the camera views the scene along the x-axis and that the objects are sorted with increasing depth, we can formally express a part's visibility as:

$$v_p^i = \textsc{area}(\boldsymbol{o}_p^i \setminus \bigcup_{j<i} \boldsymbol{o}^j) / \textsc{area}(\boldsymbol{o}_p^i), \qquad \text{s.t. } \forall_j O_x^j < O_x^i \tag{6.5}$$

where $\textsc{area}(\boldsymbol{o}_p^i)$ denotes the image area in pixels covered by the projection of $\boldsymbol{o}_p^i$. Alg. 1 gives an efficient algorithm for obtaining v_p^i for rectangular projections $\boldsymbol{o}_p^i$. As detectors tend not to respond for parts with low visibility due to the lack of occluded samples in the training data, we discard part detections when the visible area v_p^i is below a certain threshold v_{min} (in practice, $v_{min} = 0.75$). We define our multi-detector observation likelihood with explicit occlusion handling as a mixture-of-experts [50] where the experts are the part detectors and the weights are proportional to the visible area v_p^i of those parts:

$$P(\mathcal{E}|\mathbf{X}) \propto \prod_i \frac{1}{\sum_p \delta[v_p^i > v_{min}] \cdot v_p^i} \cdot \sum_p \left[\delta[v_p^i > v_{min}] \cdot v_p^i \cdot \Psi_D\left(\mathbf{d}_p^{\mathbf{a}(\mathbf{i})}\right) \cdot \Psi_G\left(\mathbf{O}^i, \Theta; \mathbf{d}_p^{\mathbf{a}(\mathbf{i})}\right) \right] \tag{6.6}$$

Here, $\mathbf{a}(\mathbf{i})$ denotes the association function that assigns candidate detections $\mathbf{d}_p^{\mathbf{a}(\mathbf{i})}$ (at most one for each part p) to every 3D object hypothesis $\mathbf{O}^i$. In case a detector is not firing despite a sufficiently large estimated visibility ($v_p^i > v_{min}$) we use a minimum appearance score to compensate missing evidence. Ψ_D and Ψ_G are defined as for the single detector likelihood, but use the associated part detector's estimate for the full object extent instead of the full body detector. Regarding the comparability of detector scores we found empirically that SVM margins on true positive detections tend to be larger for better performing detectors. This is probably due to the fact that we train all detectors on the same training set, and thus scores are implicitly normalized by scaling the SVM margin to 1. Therefore an implicit detector weighting is learned during SVM training and no further provision to balance SVM scores is required.

Algorithm 1 Efficient visible area computation for rectangular regions: $\mathbf{r}$ - rectangle for which the number of visible pixels is computed; m - maximum tested object depth.
Since the intersection and AREA can be computed quickly for rectangles, this algorithm is faster in practice than a dense pixel-wise occlusion map which is often used for arbitrary shapes.

Require: $\mathbf{O}^1, \ldots, \mathbf{O}^m$ sorted in increasing depth

function VISIBLEAREA($\mathbf{r}$, m)
  $v^r \leftarrow$AREA($\mathbf{r}$)
  for $k = 1 \ldots m-1$ **do**
    $\mathbf{o}^k \leftarrow$PROJECT($\mathbf{O}^k$)
    if $\mathbf{r} \cap \mathbf{o}^k \neq \varnothing$ **then**
      if $k \neq 1$ **then**
        $v^r \leftarrow v^r -$VISIBLEAREA($\mathbf{r} \cap \mathbf{o}^k, k$)
      else
        $v^r \leftarrow v^r -$AREA($\mathbf{r} \cap \mathbf{o}^k$)
      end if
    end if
  end for
  return v^r
end function

$v_p^i \leftarrow$ VISIBLEAREA($\mathbf{o}_p^i, i$)/AREA($\mathbf{o}_p^i$)

Multi-frame model.

In video streams it is possible to leverage evidence from adjacent frames. To that end we extend our likelihood to entire "scene tracklets" [109, Sec. 4] and define the multi-frame observation likelihood as:

$$P(\mathbf{X}_t | \mathcal{E}_{-\delta t+t:t+\delta t}) \propto \prod_{r=t-\delta t}^{t+\delta t} P(\hat{\mathbf{X}}_r | \mathcal{E}_r), \tag{6.7}$$

where $\hat{\mathbf{X}}_r$ denotes the scene configuration that has been extrapolated from $\mathbf{X}_t$ based on the camera's estimated ego-motion and assuming that object positions as well as the camera pitch vary only slowly in successive frames.

6.3.2 Inference

Hypotheses clustering.

To enable efficient inference we cluster $\mathbf{a}(\mathbf{i})$ agglomeratively into groups of possible associations. Starting from an association function $\mathbf{a}(\mathbf{i})$ that only associates full object

detections, we iteratively add associations to part detections when those overlap sufficiently for the respective object part. In each iteration we add the part detection with the highest overlap that has previously not been matched. Part detections that cannot be matched to an existing cluster lead to an additional, new cluster.

RJMCMC inference.

Inference in our model is performed by Metropolis-Hastings MCMC sampling [109, Sec. 3.1-3.2], which employs reversible jumps in order to cope with a varying number of objects in the scene. Our framework employs diffusion, add and remove proposal moves. Add proposals are adapted from the agglomeratively clustered object hypotheses, which are selected with a probability proportional to its maximum part detector score. Finally, we perform projective 3D to 2D marginalization [109, Sec. 3.3] to compute a score for each object.

6.4 EXPERIMENTAL RESULTS

We evaluate our models on two publicly available datasets: ETH-LINTHESCHER and ETH-PEDCROSS2 (see Fig. 6.5 for sample images). Both were recorded with a moving stereo camera in densely populated pedestrian zones and originally published by Ess et al. [25]. The videos are recorded at a frame rate of $\sim$14Hz and a resolution of 640×480 pixels. We only use the left camera's images as input to our monocular system[1]. ETH-LINTHESCHER is comprised of 1209 stereo image pairs with a total of 2606 annotated pedestrians. As our system with explicit occlusion reasoning is capable of detecting severely occluded pedestrians that are not contained in the original annotation, we manually extended the annotations (by all pedestrians which are at least 20% visible) to a total of 3018 pedestrians. ETH-PEDCROSS2 consists of 840 frames recorded at a pedestrian crossing and along a rather narrow sidewalk with frequent occlusions among pedestrians. As the dataset comes without annotations, we annotated pedestrians in every 4^{th} frame similar to ETH-LINTHESCHER, and included instances that are truncated by the image boundaries. Overall our annotations contain 1635 pedestrians[2]. We use the same set of parameters throughout all experiments and follow the evaluation protocol of Ess et al. [27] and consider only pedestrians with an annotation height of at least 60 pixels.

Due to the lack of 3D ground truth we project the estimated 3D models to the image plane and employ detection metrics to report full image performance as miss rate vs. false positives per image (FPPI) on a log-log scale (see Fig. 6.3). Moreover, we use the *log-average miss rate* (LAMR) for an assessment across a large range of false positive rates. We define it as the average miss-rate sampled from the lowest false

[1] We simulate yaw and speed sensor readings based on SfM results kindly provided by the authors of [25].

[2] Annotations are available at http://www.d2.mpi-inf.mpg.de.

positive rate to a false positive rate of 1 FPPI. Missing samples for high FPPI rates are filled in with the minimum miss rate of the highest false positive rate on the curve. We use equally distant samples in log-space and therefore the log-average miss rate stresses low miss rates at high precision, which is preferable for the systems' output.

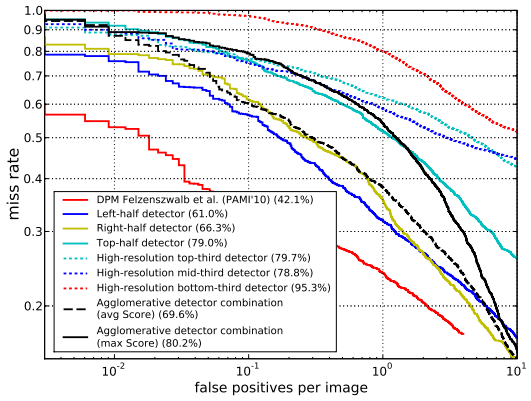
6.4.1 Results on ETH-Linthescher

We start by evaluating the performance of the different human detectors on the ETH-Linthescher sequence (see Fig. 6.3a). Firstly, we observe that the part-based full body DPM detector [34] performs best as expected with an LAMR of 42.1%. When we only use left- and right-half detectors the performance drops to 61.0% for the left detector and to 66.3% for the right detector, respectively. The performance for the upper body top-half detector is even worse with an LAMR of 79.0%. This drop in performance may be explained by the missing discriminative evidence of the legs as lower object boundary. The high-resolution top-third pedestrian detector and mid-third detector roughly perform at the same level and achieve an LAMR of 79.7% and 78.8%, respectively. The missing recall is mostly due to pedestrians that appear at small scales that cannot be scanned by this detector. The bottom-third (feet) detector performs worse than the top-third and mid-third detectors and achieves an LAMR of 95.3%. When we combine all detectors by agglomerative clustering as described in Sec. 6.3.2, the combination achieves an LAMR of 69.6% when we use the average score of all detectors for the cluster and 80.2% when we use the maximum score. While the clustering of detector hypotheses yields an unsatisfying false positive rate, the achieved minimum miss rate (11.2%) is promising and lower than for all stand-alone detectors. We thus use this detector combination as input and baseline for our models that employ explicit occlusion reasoning. As we will see they successfully improve performance compared to systems with full-body detectors only.

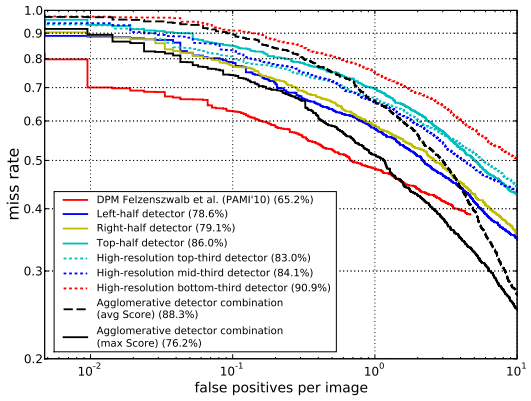
Next we analyze our model's performance with explicit occlusion handling (Fig. 6.3c). As a baseline we run our model with evidence from a single frame and without occlusion reasoning. With this setup only a small improvement over agglomerative clustering can be achieved (LAMR 59.7%). When extending the evidence over multiple frames and performing scene tracklet inference, we achieve an LAMR of 52.2%. Most importantly, however, an even larger performance gain to an LAMR of 42.2% is accomplished when using our newly proposed explicit 3D occlusion reasoning scheme. We note that this model already outperforms the standalone full body detector. When additionally using scene tracklets and thus the full model, we achieve a further improvement to an LAMR of 37.3%.

Finally, Fig. 6.3e compares our model to other state-of-the-art approaches[3]. The modular stereo system of Ess et al. [27] performs at an LAMR of 57.1%. Our previously proposed system with a single full body detector [109] achieves an LAMR of 51.8%. We note that for both systems the minimum miss rate saturates at 30%-40%.

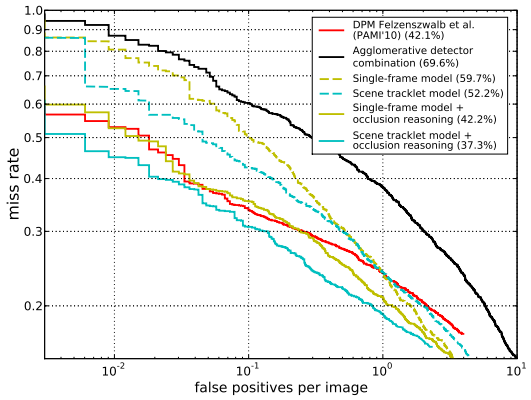
[3]The authors of [27] kindly provided us with their latest result (for our annotations).



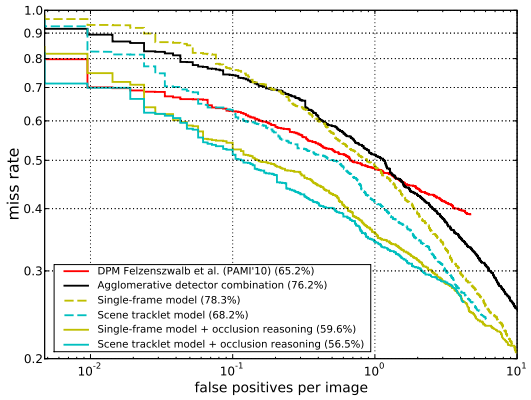
(a) Detector performance (ETH-LINTHESCHER)



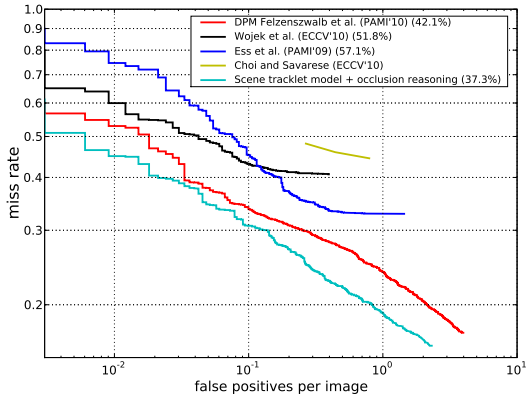
(b) Detector performance (ETH-PEDCROSS2)



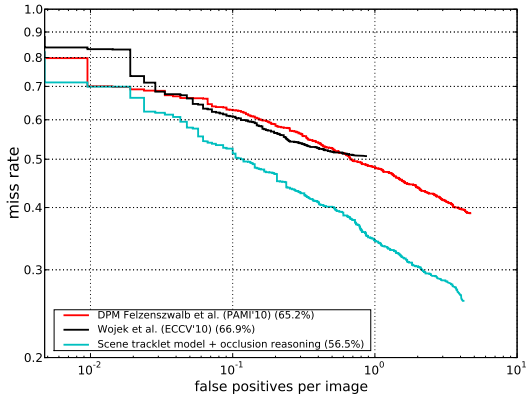
(c) Scene models (ETH-LINTHESCHER)



(d) Scene models (ETH-PEDCROSS2)



(e) Comparison to state-of-the-art (ETH-LINTHESCHER)



(f) Comparison to state-of-the-art (ETH-PEDCROSS2)

Figure 6.3: The first column shows results for the ETH-LINTHESCHER sequence, the second column for the ETH-PEDCROSS2 sequence. Percentages in the legends indicate the log-average miss rate (see text for definition). The first row shows the standalone detector performance. DPM [34] performs overall best, but detectors for body parts are able to achieve a lower miss rate at the cost of lower precision. The second row depicts our models' performance for different configurations. The last row compares this performance to other state-of-the-art methods. Our models outperform the stand-alone DPM detector [34], the stereo system by Ess et al. [27], our previous monocular single full-body detector system [109] and the monocular system by Choi & Savarese[8].

Our model on the contrary achieves an LAMR of 37.3% with a substantially lower minimum miss rate of only 16.0% while also improving the false detection rate for all miss rates. Choi and Savarese [8] report three points on the recall vs. FPPI curve, which are not competitive compared to our model; the miss rate is about 20 percentage points higher at the same error rates. We note that [8] reports performance for the original annotations, which do not include all occluded pedestrians. Hence, the performance on the modified annotation set may be slightly worse. Overall, our full model with explicit occlusion reasoning and scene tracklets outperforms three state-of-the-art approaches. In particular it achieves the highest recall among all models and reduces the error rate along the entire curve. Fig. 6.5 compares our full model on some sample scenes to competing state-of-the-art approaches.

6.4.2 Results on ETH-PEDCROSS2

Next, we turn to the more difficult ETH-PEDCROSS2 sequence, which contains more occluded pedestrians. Again, we start by analyzing the detectors' performance alone (Fig. 6.3b). Similar to ETH-LINTHESCHER, DPM [34] yields the best standalone detector performance with an LAMR of 65.2%. The next best performance again is achieved by left- and right-half detectors with 78.6% and 79.1% LAMR, respectively. The top-half detector performs at 86.0% LAMR. For the high-resolution detectors the top-third (83.0% LAMR) and mid-third detectors (84.1% LAMR) again perform better than the bottom-third detector (90.9% LAMR). Interestingly, the agglomerative detector combination with average scores (88.3% LAMR) performs worse than when the maximum score is used (76.2%). We conjecture that low scores of the full-body detector on the occluded samples lower the average score on this dataset.

When we apply our single-frame 3D scene model (Fig. 6.3d), the performance of the agglomeratively clustered detector combination is improved only slightly to 78.3%. Again, explicit occlusion reasoning improves the results more (59.6% LAMR) than the scene tracklet formulation (68.2% LAMR). However, as for the previous sequence, the best performance is achieved when both tracklet inference and explicit occlusion reasoning are performed (56.5% LAMR).

We finally compare our model to our previously proposed model [109], which only uses a single full body object detector (Fig. 6.3f). In terms of LAMR our full model outperforms this segmentation supported model (LAMR 66.9%). It is also instructive to go back to the DPM full body detector, which is also clearly outperformed by our model. In particular our full model achieves a substantially lower minimum miss rate (26.0%) compared to the two baselines, which achieve 39.0% [34] and 50.7% [109]. For our full model we have also analyzed false positive failures that occur with a high score. Fig. 6.4 shows the two highest scoring detections that are due to false partial detections and incorrectly supported by occlusion reasoning.

For this dataset we additionally analyzed the detection performance on partially occluded pedestrians. To that end we annotated all partially occluded pedestrians

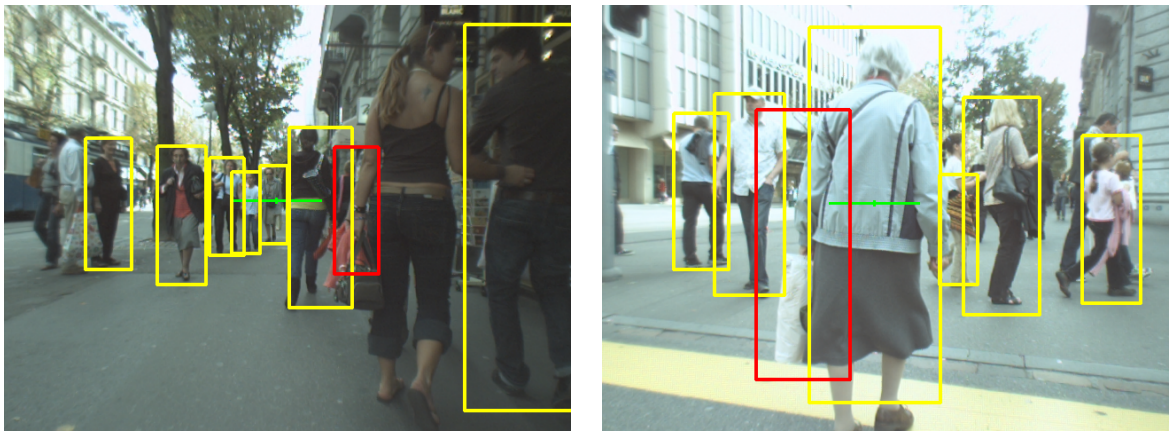


Figure 6.4: First (left) and second (right) false positive detection on the ETH-PedCross2 sequence for our proposed model with occlusion handling. Both false positives (red bounding boxes) are due to false detections of a left- or right-half detector and are strengthened by occluding true positives. The false detection on the left actually detects an occluded true pedestrian, but with a too large scale. This detection also suppresses the true detection on the pedestrian in the close range.

and performed the evaluation restricted to these instances. Overall 1052 pedestrians were marked as partially occluded out of which DPM [34] detected 40.7%. Our previous approach without explicit occlusion reasoning [109] was able to detect only 19.2%. This low recall compared to the standalone detector is mostly due to the tracklet formulation, which tends to drop detections that are partially occluded in at least one frame of a tracklet. The proposed model, on the other hand, can solve this shortcoming using an explicit 3D occlusion reasoning and achieves a recall almost three times better (55.0%) on occluded pedestrians. The algorithm's runtime is dependent on the number and density of objects in a scene; the current C++ implementation runs for about two seconds on average per frame on recent hardware.

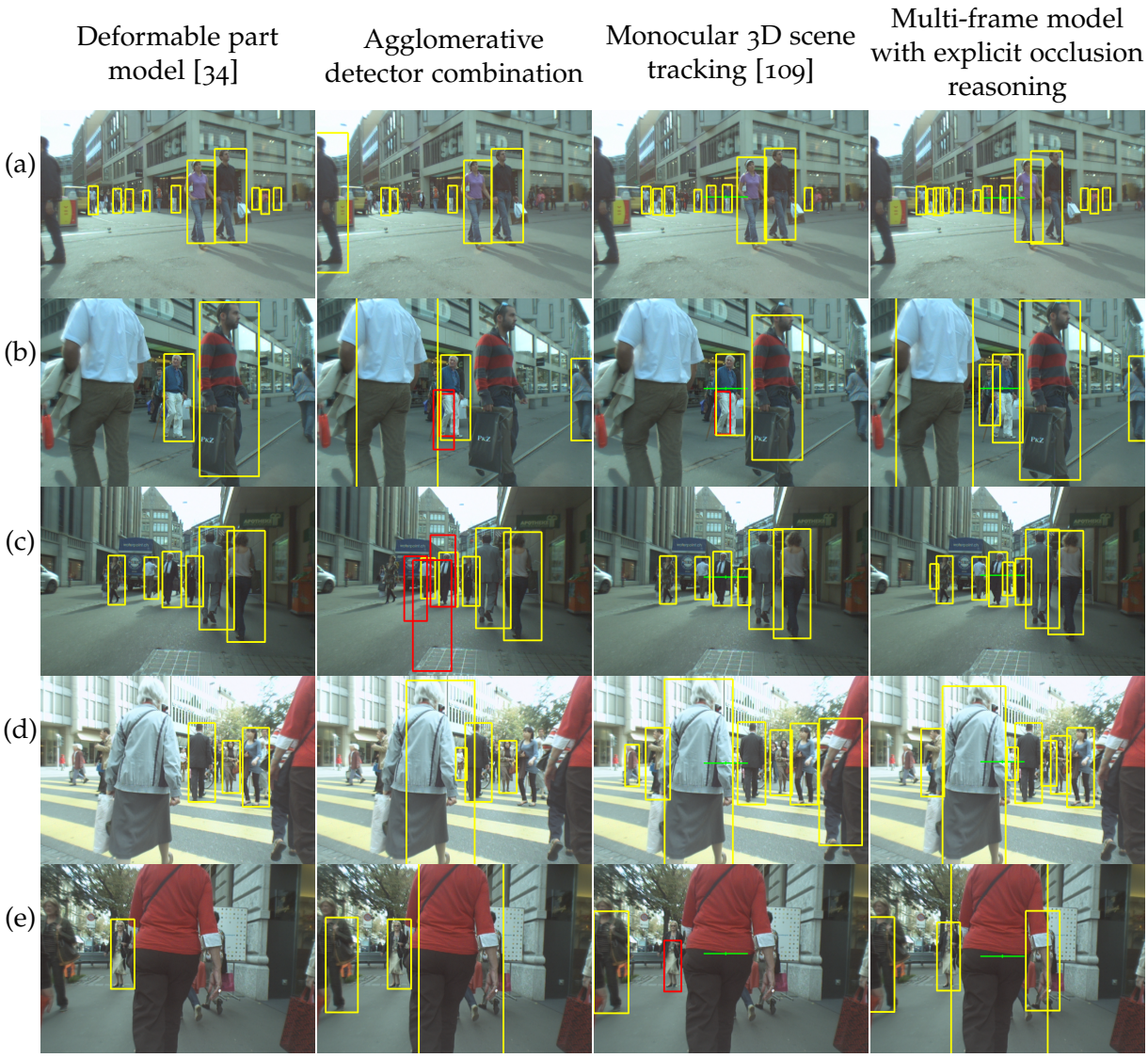


Figure 6.5: The first three rows show sample scenes for ETH-LINTHESCHER, the last two rows for ETH-PEDCROSS2. All results are depicted at a constant error rate of 1 FPPI. Yellow bounding boxes are true positives, red boxes are false positives. Both DPM and the model proposed by Wojek et al. [109] are not able to handle occlusion and object truncation at the image boundary well. On the contrary our model shown in the rightmost column is able to detect occluded pedestrians as well as pedestrians very close to the camera.

6.5 DISCUSSION AND CONCLUSION

We introduced a model for multi-object tracking from a moving platform that combined 3D scene tracking, full object and object part detectors, and explicit 3D object-object occlusion reasoning to also handle objects that are partially occluded for long durations of time or never fully visible at all. As our experiments with multi-people tracking have shown, our model with explicit occlusion reasoning is capable of robustly detecting occluded and truncated pedestrians by strengthening weak evidence obtained from partial human detectors through the accumulation of geometric scene constraints and by evidence obtained over multiple frames. The proposed model outperforms similar monocular approaches [8, 109] without occlusion reasoning, as well as a stereo-based system [27], and is able to obtain a substantially higher recall than these competing approaches. Also, our approach outperforms state-of-the-art part-based detectors [34].

7 TOWARDS A BETTER UNDERSTANDING OF FEATURE COMBINATION AND EVALUATION

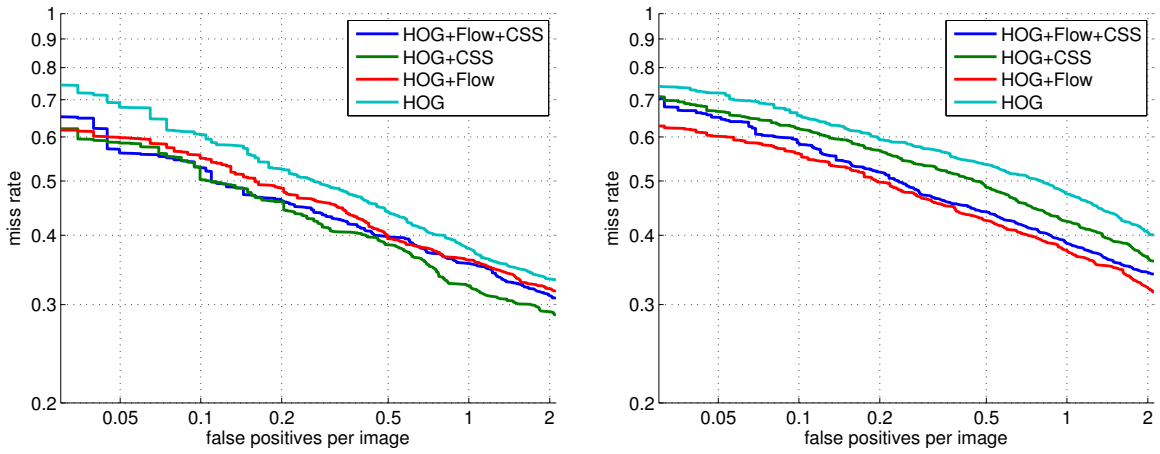
Contents

7.1 INTRODUCTION

When we introduced CSS in chapter 4, it seemed that it only provides moderate gains, focused in the high-recall regime of the detector. Its benefits were also visible especially in the static image setting without motion features, only resulting in a small gain when motion features were included. In this chapter, we will demonstrate that this was due to a lack of training data and proper training procedures. With the multi-stage training technique from chapter 5, we can increase the available training data and so demonstrate that using CSS provides significant gains also when using motion information and that it increases recall over the complete range of precision. We also demonstrate that quantitative results at very low FPPI rates are unreliable and that caution must be taken when comparing strengths and weaknesses of detectors.

7.2 A RE-EVALUATION OF COLOR AND COLOR SELF-SIMILARITY

In chapter 4, we introduced a new feature, color self-similarity (CSS), that captured pairwise statistics of color histograms in the detector window, and showed that



(a) ETH-Loewenplatz, threshold 0

(b) TUD-Brussels, threshold 0

Figure 7.1: CSS changed its performance contribution after reimplementing. The combination of HOG, Flow and CSS does not result in the best performance, as expected.

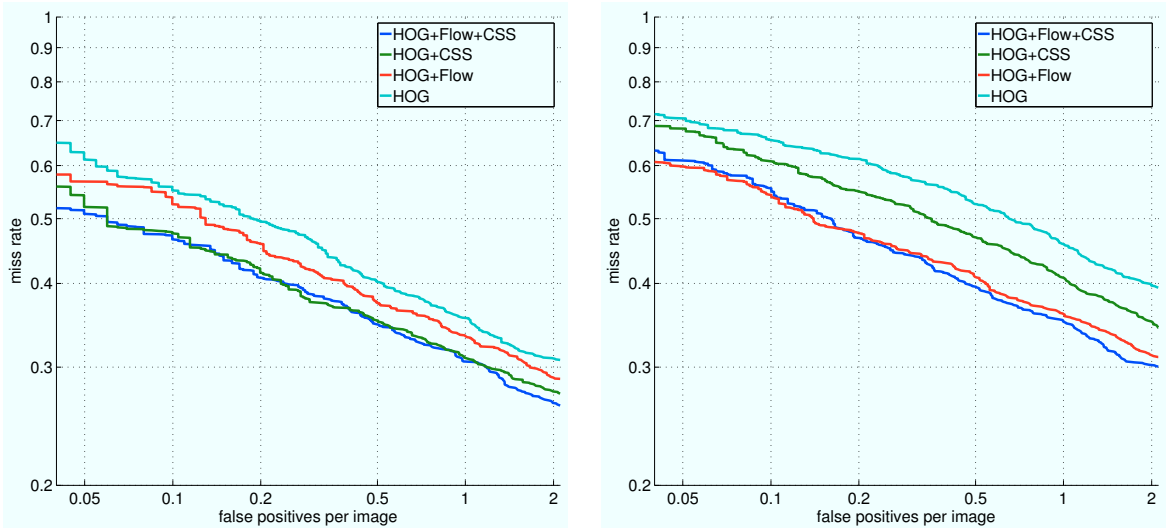
it can improve detection rates on TUD-Brussels (see figure 4.4a) and the Caltech Pedestrian database (see figure 4.5).

However, measured against the high dimensionality of the feature and its high runtime cost, the performance gains were small and unsatisfying. This becomes especially apparent when keeping in mind that intuitively this feature should perform far better since it captures a kind of information not yet present in the feature descriptor consisting of encoded gradients (HOG) and optical flow (HOF). It also uses proven techniques (histogram binning, spatial interpolation and interpolation between histogram bins) to make the feature robust. The unsatisfying performance was the prime reason CSS was not used in our later publications.

An additional difficulty we encountered with CSS was that it required careful tuning until it worked as shown in chapter 4. We later re-implemented this feature in order to make the feature computation faster and more flexible (e.g. easily allowing for tuning the amount of spatial smoothing). After reimplementing, we tuned the (new, distinct) parameter set to result in the same performance when trained and tested on INRIA Person (used as a kind of validation set here). However, training on TUD-MotionPairs and testing on TUD-Brussels (our setting in chapter 4) or ETH-Loewenplatz resulted in an unexpected outcome. This happened despite the fact that we use the same classifier (an intersection kernel SVM) and the same feature combination scheme as in chapter 4.

What we expect when we combine HOG, Flow and CSS is that HOG+CSS and HOG+Flow perform better than HOG alone, and HOG+Flow+CSS performs even better. As can be seen in figure 7.1a and figure 7.1b, this is not the case: The relative order is not what we expect, and it is different for the two test sets.

While HOG+Flow and HOG+CSS perform better than HOG alone on both



(a) ETH-Loewenplatz, threshold -1 (b) TUD-Brussels, threshold -1

Figure 7.2: Setting the retraining threshold to -1 stabilizes performance. The result is closer to the expected result than using a threshold of 0 (figure 7.1). Using HOG+Flow+CSS no longer performs noticeably worse than HOG+Flow or HOG+CSS.

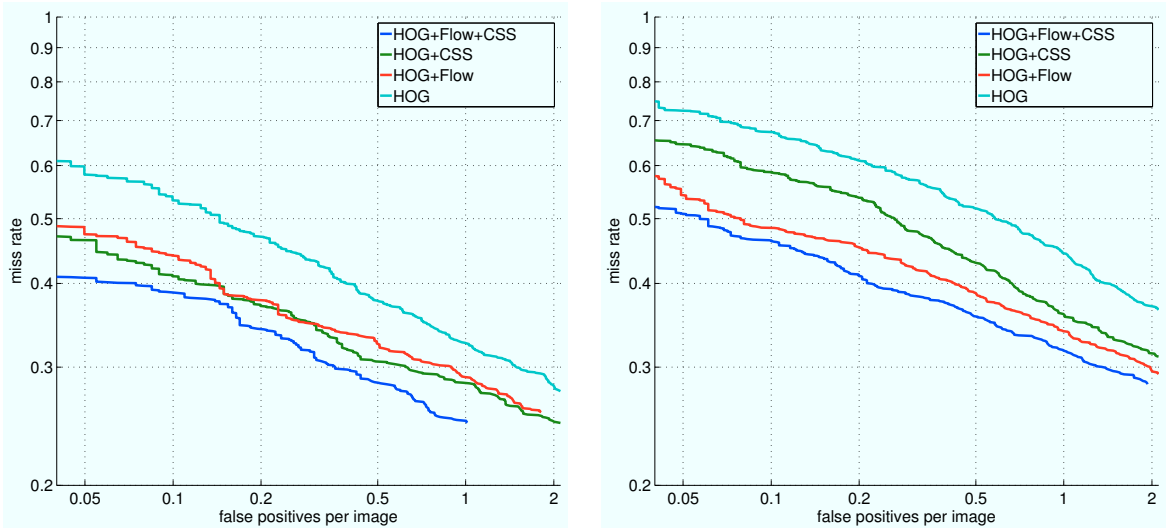
datasets, on TUD-Brussels HOG+Flow is the top performer and adding CSS results in a noticeable drop in performance. On ETH-Loewenplatz, HOG+CSS performs best and adding Flow decreases performance.

From exploring the causes of this unexpected outcome, there are multiple lessons to be learnt:

Lesson 1 In addition to the number of retraining rounds, the threshold for "hard" negative samples plays an important role during retraining, *simply using misclassified samples does not suffice*.

Lesson 2 The training set used in chapter 4, TUD-MotionPairs, is far too small for training a detector using HOG+Flow+CSS as a feature set, and by expanding the training set we can show that *the improvements that can be gained from using CSS are far larger than indicated in chapter 4*.

Lesson 3 Using the enlarged training set, we find, contrary to our conclusion in chapter 4, that *raw color histograms actually can lead to an improvement in detection rates*. However, this does not reduce the usefulness of CSS as it indeed contains information that is complementary to raw color.



(a) ETH-Loewenplatz (b) TUD-Brussels

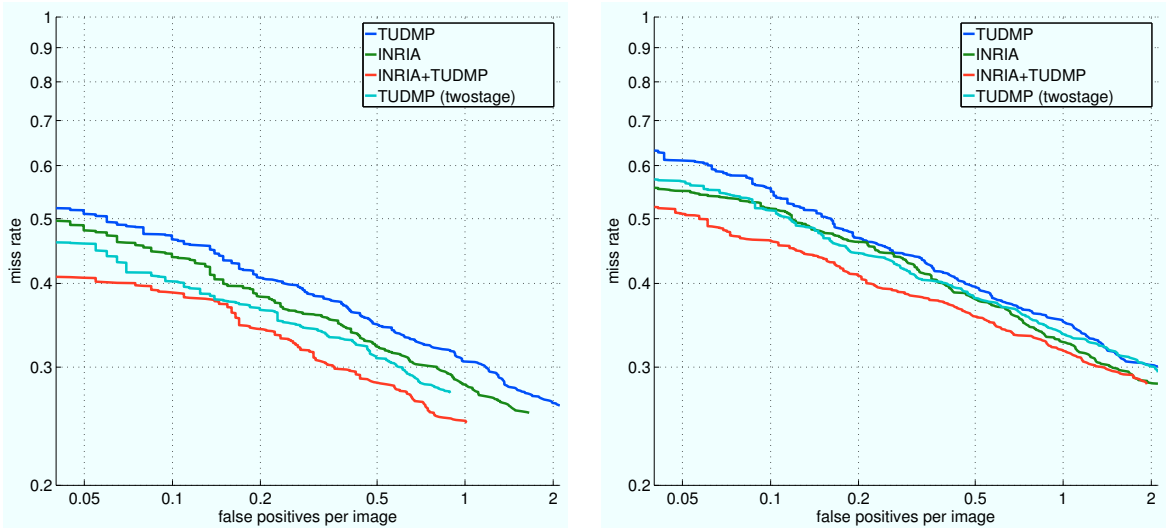
Figure 7.3: Performance when training on the combined dataset. For almost all cases, the performance noticeably improved in comparison to figure 7.2.

7.2.1 The Choice of Hard Samples During Retraining

During the retraining phase we have to make a choice what to classify as a hard sample. To make this choice consistent across classifiers, we opted to label a sample as a hard negative when it was on the "wrong" side of the learned decision boundary (effectively assuming a 0-1-loss). This means a threshold of 0 for SVMs and 0.5 for boosting-type classifiers (which used a sigmoid to produce confidence values).

However, for SVMs the more natural choice is a threshold of -1, which means that samples with a nonzero loss (meaning they would be support vectors with the current SVM) get tagged as hard negatives. In earlier experiments (without CSS) we evaluated the effects of using different thresholds when using SVMs, and did not find significant performance differences between classifiers trained with different thresholds for hard samples. In our current situation however, using the new implementation of CSS, setting the threshold to -1 stabilizes the performance, as can be seen in figures 7.2a and 7.2b. For ETH-Loewenplatz, using a threshold of 0 resulted in HOG+CSS performing better than HOG+Flow+CSS (figure 7.1a). With a threshold of -1, both methods are on par (figure 7.2a). The results for TUD-Brussels look similar, as HOG+Flow performs better than HOG+Flow+CSS using a threshold of 0 (figure 7.1b) and both methods produce similar results using a threshold of -1. So, for both test sets using the full feature set no longer results in suboptimal performance, however it is still not the unique best feature set.

The loss functions we used for boosting (exponential loss for AdaBoost and logistic loss for MPLBoost) are never zero, so there is no natural threshold in this sense for those classifiers. How to choose the threshold here in order to get a stable



(a) ETH-Loewenplatz (b) TUD-Brussels

Figure 7.5: Performance for different training sets using HOG, Flow and CSS. The joint training set (INRIA+TUDMP) performs better than training on INRIA Person (INRIA) or TUD-MotionPairs (TUDMP) alone. Using the two-stage training procedure with TUD-MotionPairs as the training set in both stages (TUDMP-twostage) also improves performance in comparison to using the "normal" training procedure on this dataset (TUDMP).

performance remains an open question.

7.2.2 Expanding the Training Set

Another part of the explanation for the unstable performance is the high feature dimension that one obtains when combining HOG, HOF and CSS. Learning a classifier in a 14428-dimensional feature space with only 1787 positive samples (3574 with mirroring) is very prone to overfitting. However, simply adding more training samples is not trivial, as we have seen in chapter 5, where using TUD-Aux as a training set for motion and appearance resulted in far inferior performance compared to TUD-MotionPairs, even when used in addition to it. Tried and proven training sets like INRIA Person don't have motion information associated so they are unsuited for our motion-enhanced detector using our current training scheme.

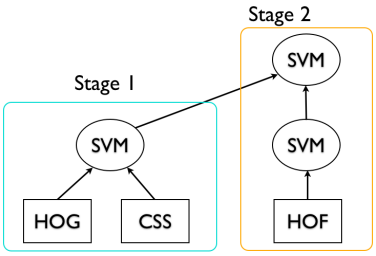


Figure 7.4: Two-stage training

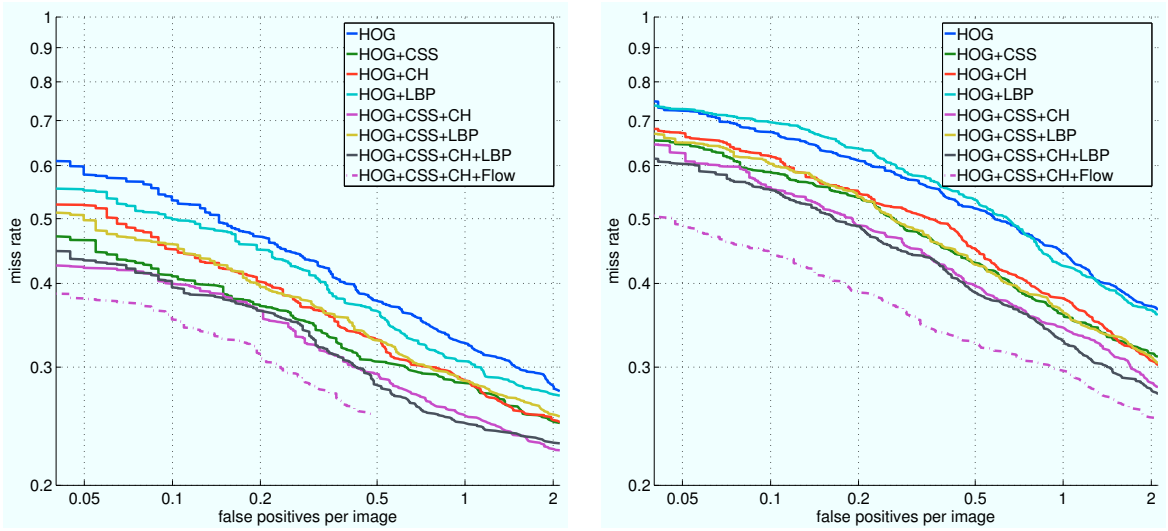
However, we can reuse the combination scheme from chapter 5 (figure 7.4). To alleviate the problem of too few training samples, we combine INRIA Person and TUD-MotionPairs as training sets. Since there is no motion information

available for INRIA PERSON, we utilize a multi-stage training method when we want to use motion features: In the first stage, we train a classifier using appearance-only features (e.g. HOG and CSS) on both TUD-MOTIONPAIRS and INRIA PERSON (combining both the positive and the negative training set), complete with multiple retraining rounds. In the second stage, we train a classifier on the Flow feature using TUD-MOTIONPAIRS, holding the appearance-classifier fixed. Jointly with this, we learn a second-level SVM that determines the relative weights between the appearance- and the motion-classifier. This is done via 5-fold cross validation to prevent overfitting.

In figures 7.3a and 7.3b one can see the beneficial effect of this enriched training set. Not only is the relative ordering of algorithms as we expect for both test sets, but also the relative difference between algorithms is more pronounced and stable over the complete range of false positive rates. In figure 7.3 the detector using HOG+Flow+CSS clearly produces the best results for both test sets. This is a clear improvement to figure 7.2, where HOG+Flow (figure 7.2b) or HOG+CSS (figure 7.2a) produced equally good results.

The only case where combining the training set hurts performance compared to training on TUD-MOTIONPAIRS is where HOG alone is used as a feature. The cause for this is probably the prevalence of front views in INRIA PERSON, which results in a HOG template that is biased towards those, while the test sets also have a large amount of side views. When adding CSS, the classifier benefits more from the increased training set size because of the feature dimension. The (relative) performance gain for HOF can be explained by the fact that this training procedure is encouraging the complementarity of appearance and motion cues: HOF works, by design, best on side views (because those tend to move independently from the rest of the scene), and it is trained on TUD-MOTIONPAIRS, which focuses more on side views than INRIA PERSON.

In figures 7.5a and 7.5b, the performance for different training sets is displayed, using HOG+Flow+CSS as a feature. The curves marked INRIA are obtained by using the same two-stage training procedure as for the joint training dataset, but with only INRIA PERSON for training. The motion part of the detector is always trained on TUD-MOTIONPAIRS. One can clearly see that using the combined dataset for training results in the best performance on both datasets. An interesting result is that applying the two-stage training using only TUD-MOTIONPAIRS substantially improves performance in this case, especially on ETH-LOEWENPLATZ. The cause for this is that the stronger regularization implied by this scheme lessens the effect of overfitting.



(a) ETH-Loewenplatz (b) TUD-Brussels

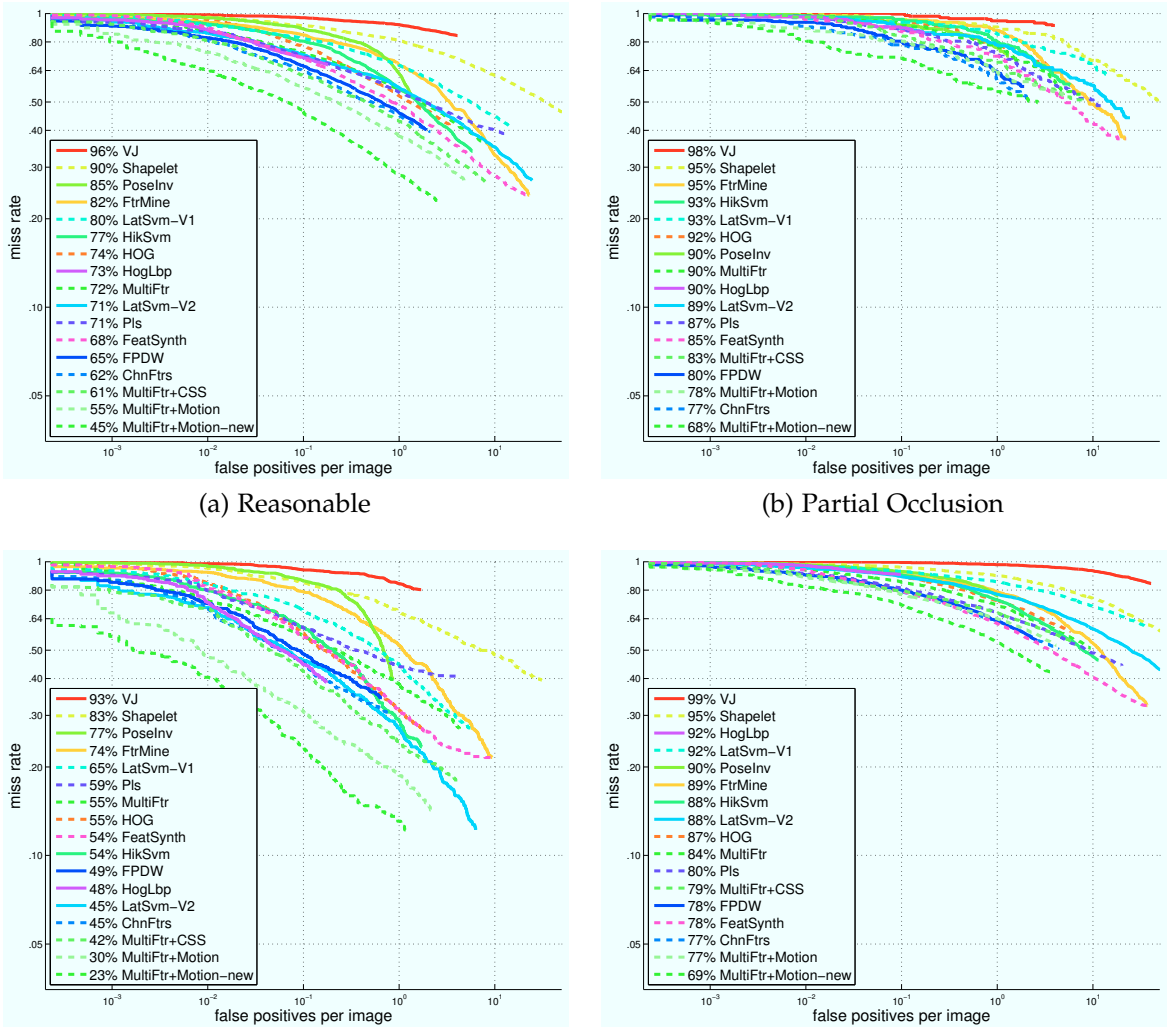
Figure 7.6: Re-Evaluation of Local Binary Patterns (LBP) and Color Histogram (CH) features

7.2.3 Re-Evaluating Discarded Features

In chapter 4, we discarded the direct usage of color histograms and local binary patterns (LBP) because including them was hurting performance[1]. As the cause of that was suspected to be different feature statistics in training and test sets, a re-evaluation of this statement using our new training set is in order.

Figure 7.6 shows the results of this re-evaluation for selected combinations of HOG, CSS, CH (the HSV color histograms that form the base of CSS) and LBP. One – given the results of chapter 4 – surprising result is the beneficial effect of adding raw color histogram features, which consistently improves the detection performance on both datasets in every combination of features (compared to the same combination without CH). What did not change is the unstable performance of LBP: While sometimes – e.g. in the HOG+LBP setting on ETH-Loewenplatz – helpful, there are cases where it has a negative impact (HOG+LBP on TUD-Brussels). Also, while not reducing performance when combined with HOG+CSS+CH, it does not consistently improve performance either. Another fact that is not expected is that the best static image feature detector (using no motion or stereo information) performs slightly better than our best detector from chapter 5 (which is using appearance, motion and stereo) on ETH-Loewenplatz! The dashed line in 7.6 is the performance that we obtain if we add HOF to the HOG+CSS+CH detector, resulting in a nice performance boost, and underlining the beneficial effect of using classifiers from different sources.

[1] In the case of LBP, it improved performance on INRIA Person like in the original work of Wang *et al.* [105], however it reduced performance in our target setting – training on TUD-MotionPairs and testing on TUD-Brussels.

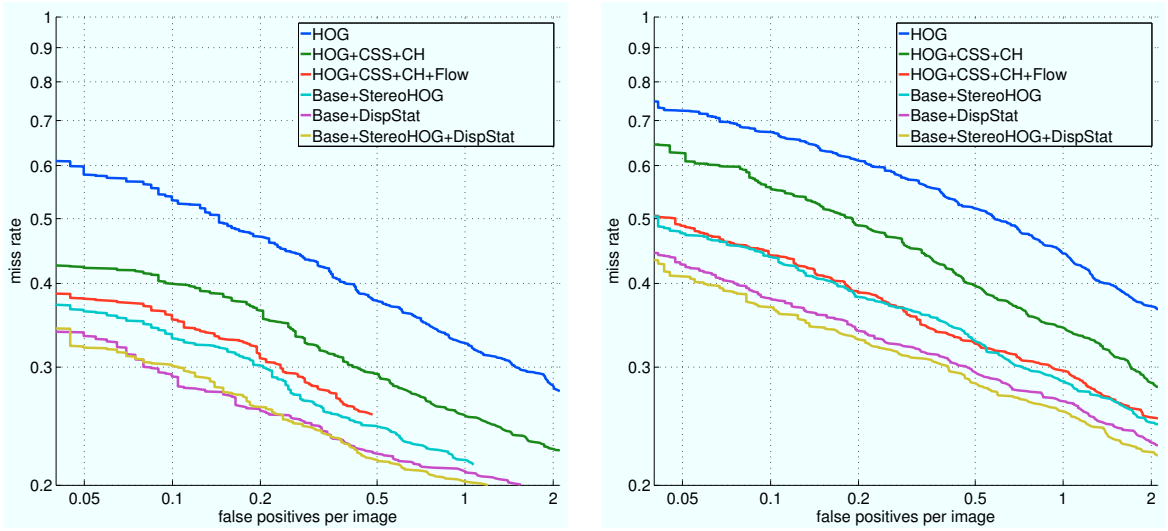


(a) Reasonable (b) Partial Occlusion

(c) Near (d) Medium

Figure 7.7: Evaluating HOG+CSS+CH+HOF on Caltech-Pedestrians

We also evaluated this new feature combination (HOG+CSS+CH+Flow) on Caltech-Pedestrians, in the same setting we used in chapter 4. The results can be seen in figure 7.7. MultiFtr+Motion is the detector from chapter 4 (which had the best overall performance in the evaluation of Dollár *et al.* [19]), MultiFtr+Motion-new is our new detector. The improvements are vast – e.g. on the "reasonable" subset, the LAMR is 10 percentage points lower. In almost all evaluation subsets there is a visible gap between our new detector and the algorithms surveyed in Dollár *et al.* [19] (e.g. in the "partially occluded" subset, performance differences between two adjacent detectors are 0–3pp LAMR, while MultiFtr+Motion-new performs 9pp better than the next contender, ChnFtrs[16]. In the near setting, the differences between MultiFtr+Motion and MultiFtr+Motion-new are smaller, but still very noticeable. Since the detector works already very good in this setting (with a LAMR of 30%), our new detector "only" delivers an improvement of 7 percentage points. The drastic



(a) ETH-Loewenplatz (b) TUD-Brussels

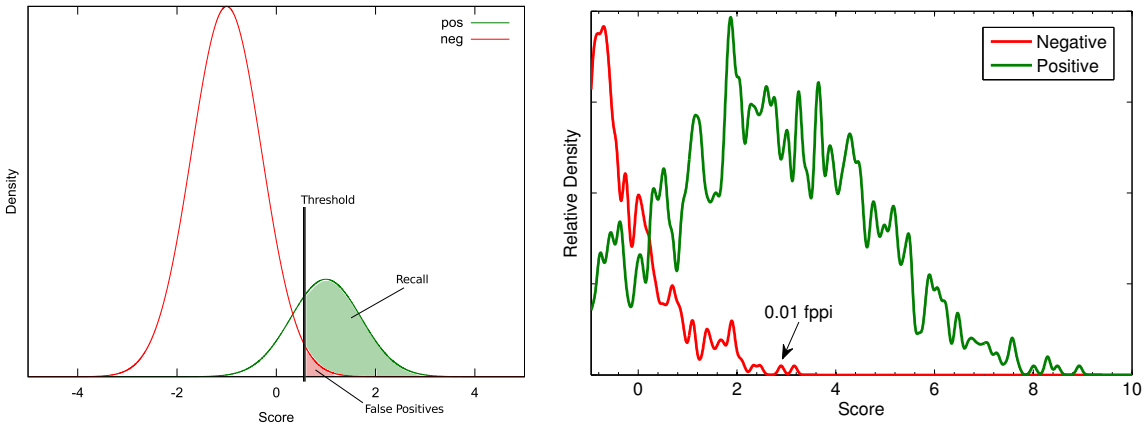
Figure 7.8: Combination of HOG+CSS+CH+Flow (abbreviated "Base" in the legend) with the stereo features of chapter 5

differences seen on this dataset again underline the importance of good features and good data to train classifiers on those good features.

Unfortunately, when adding stereo features, one can see diminishing returns[2]: The same features – StereoHOG and DispStat – that resulted in an improvement in over 10pp recall at a given false positive rate in chapter 5 – "only" improve performance by about 5pp for our new detector. As a sanity check, we replaced the Appearance+Flow component from the best detector in 5 with our new detector and got about the same performance improvement by the stereo components trained back then, so the diminishing returns are not an effect of a faulty training procedure now. Especially StereoHOG seems to struggle in this setting, maybe because it captures a similar kind of information as CSS – an implicit foreground-background segmentation (albeit from a different source) – so combining them won't be as beneficial as the sum of the improvements the features provide on their own. However, this still means that at a given recall rate the false positive rate is reduced by a factor of two, which is still a noticeable improvement.

Even although we were able to show a significant jump in performance using our new method, there is still room for improvement here, because our method of training the stereo detector has an unfortunate side-effect: We take a pre-trained detector (Appearance+Flow) trained on a different dataset, and learn the stereo component and the top-level SVM (that is responsible for weighting the components) on the same dataset. While this is done via cross-validation, there is still a dataset-inherent bias [90], which means that the top-level SVM thinks the stereo component

[2]The same is the case with classifier combination, which does not yield any noticeable improvement anymore.



(a) Score threshold

(b) Score density for HOG+HOF on ETH-LP

Figure 7.9: Measuring performance at 10^{-2} FPPI is unstable

is better than it is expected to be on a different training set. There is no easy way out of this except training the top-level SVM on a separate validation set, however, datasets usable for training with motion information and a second camera view to compute stereo features are rare, which was the original reason to introduce TUD-Aux in the first place.

7.3 STABLE AND UNSTABLE REGIONS OF FPPI CURVES

A frequent comment to our work [104], which is the basis for chapter 5, was that we changed the lower threshold for the FPPI plots, "hiding" the performance of our detectors at "low" FPPI rates. The reason for this is that, for small datasets like ETH-Loewenplatz, measurements at FPPI rates like 10^{-2} are *inherently unstable and extremely dependent on dataset-specific parameter tuning*.

The cause of this can be seen in figure 7.9. One should recall how miss rate/FPPI curves are generated (figure 7.9a): A detector produces scores for instances of positive and negative samples, with distinct score distributions for each class. For each fixed score threshold, we get values for miss rate and FPPI (or precision and recall if we desire a precision-recall plot), and those data points are represented in the performance plot.

Let us now look at the distribution in a real setting (figure 7.9b). Displayed is the score distribution for true and false positives of a detector using HOG and HOF with an intersection kernel SVM on ETH-Loewenplatz. The discrete score distribution was convolved with a gaussian to visualize the distribution. The arrow points at a peak generated by a single detection. For simplicity, we will use $\frac{2}{201} \approx 0.00995$ (instead of 0.01) FPPI as the reference point. As ETH-Loewenplatz consists of 201 annotated test images, this corresponds to exactly two false positives over the whole data set.

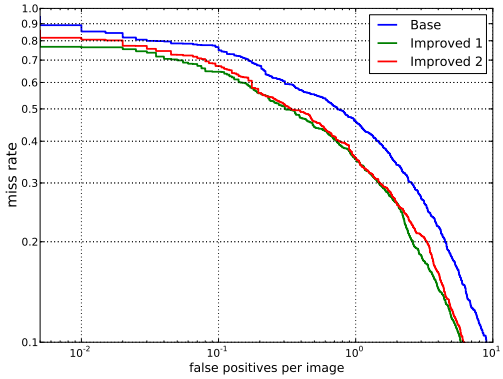


Figure 7.10: Results for synthetically generated data

Class	Attribute	Frequency
Pos.	Occluded	20%
Pos.	Plain clothing	30%
Pos.	Bad lighting	30%
Pos.	Odd pose	15%
Neg.	Cluttered background	50%
Neg.	Body part	5%

Table 7.1: Frequencies of attributes in the synthetic dataset

The measure of instability we will use is the influence that the score of a single false positive (the second strongest one) has on the recall for this false positive rate. The range of recall that can be covered by changing the score of the second false positive is the fraction of true positives that lies between the first and third false positives.[3]

In figure 7.9b, one can directly see that our reference FPPI rate (the second false positive from the right) coincides with a high-density region for the positive class. As we are at the extreme end of the negative distribution, negative samples are sparse and so the scores of the first and third false positives are at a noticeable distance. Because of this, small changes in the position of the second false positive result in big changes in recall.

This means that by tuning our detector to the second false positive (for better or worse), we could cover a range of 10 percentage points of miss rate at our reference point. However, the situation is even worse since the instances at the tail are often correlated (in this case, the first two false positives are the same object in adjacent frames), making the score even more unstable.

Because of this, while it makes sense to evaluate at low FPPI rates on datasets like Caltech-Pedestrians which have many images, so that a low FPPI rate still corresponds to enough false positive instances, one has to be extremely careful when using this measure on datasets with few images (like ETH-Loewenplatz and TUD-Brussels). We did this in chapter 6 because for integrated systems, the performance at low FPPI rates is what one is interested in, however one always has to keep in mind that specific quantitative results may be *very* deceptive.

[3] If the score is modified to be higher than the one of the first false positive, the first becomes the second false positive and so determines the new threshold at the reference false positive rate. That likewise happens for the third false positive.

7.4 QUALITATIVELY ANALYZING DIFFERENCES BETWEEN TWO DETECTORS

When comparing object detectors, the usual method of comparing the performance of two or more detector systems is plotting their miss rate (or, equivalently, recall) against precision or false positive rate (per image). This results in a plot like shown in figure 7.10.

From this plot, one concludes that the methods drawn in red and green do a better job than the method drawn in blue. A natural question when one method performs better than another is "What is the cause for this, what does this method do right that the other does wrong?" In order to answer this question, it is often suggested to analyze e.g. the added recall at a fixed false positive rate.

When this is done, one usually sees patterns in the detections that were missed by one detector but recognized correctly by the other detector. The naive conclusion would be that those patterns are representative of the better detectors' strengths. However plausible this conclusion is, it is generally unjustifiable. We will use a counter-example from synthetic data[4] to show that caution must be exercised when drawing conclusions from this approach. Reasons for this are:

- The outcome of this procedure is heavily dependant on the (arbitrarily chosen) reference point – the results for 0.1 FPPI will very likely look different from those at 1 FPPI.
- It is possible that the conclusions for *all* reference points are wrong, so there is no way to magically choose the "right" threshold.

7.4.1 Technical Setup of the Synthetic Example

For the synthetic example, we attach to each sample 44 randomly assigned binary attributes. Some attributes with arbitrarily chosen[5] names are shown in table 7.1. The attributes are disjoint for the positive and negative classes (as e.g. the "odd pose" attribute does not make sense for a negative sample). In our example, 1500 positive and 15000 negative samples are generated, mirroring the asymmetric distribution one encounters in a detection setting, where there are far more negative than positive samples.

The contribution of attribute k to the score of a positive sample is modelled to be normally distributed with mean μ_k^a and standard deviation σ_k^a if the attribute is active ($x_k = 1$). If it is inactive ($x_k = 0$), the parameters are μ_k^i and σ_k^i respectively.

[4]Unfortunately, a "real" dataset with detailed attributes for each instance does not exist yet. Creating such a dataset is a challenge in itself because it is not known what the relevant attributes are, and there is the danger of subconsciously projecting expectations onto the dataset while labeling.

[5]The names have no meaning in this synthetic example, they are used for easier reference. The attributes could have been named A, B, C, ... as well.

An attribute that is beneficial for the classification of a positive sample has $\mu_k^a > \mu_k^i$. The detector scores in our sample are modeled to be a sum of contributions from the attributes. This results in

$$\mathcal{P}(x) = \sum_k x_k \mathcal{N}(\mu_k^a, {\sigma_k^a}^2) + (1 - x_k)\mathcal{N}(\mu_k^i, {\sigma_k^i}^2) \tag{7.1}$$

Negative samples are treated likewise with parameters $\tilde{\mu}_k^{a,i}$ and $\tilde{\sigma}^{a,i}$ instead of $\mu_k^{a,i}$ and $\sigma^{a,i}$, respectively.

For our "base" classifier, we randomly[6] generate the parameters to the normal distributions, except for a few named attributes (table 7.1) where we set the parameters by hand. This is done in order to ensure that the names fit the attributes influence – e.g. bad lighting should make the task harder. The priors for the means are selected so that a "reasonable" score distribution results (with higher scores for positive samples than for negative samples, but overlapping score distributions for the two classes).

As with "normal" detection, we threshold the samples at varying scores to get our results (figure 7.9a). For a given threshold, positives samples that have a score higher than the threshold are true positives, the others are false negatives, the distinction for negatives is likewise. Each threshold results in a point on the results plot. The results of the base classifier are shown in blue in figure 7.10[7].

We create two improved detectors by making the following adaptations, starting from the base detector:

- For the detector titled "Improved 1" (figure 7.10, green) we reduced the negative influence of the attribute we named "Cluttered background". This is done by decreasing $\tilde{\mu}_0^a$.
- For "Improved 2" (figure 7.10, red) the negative impact of the presence of the "Odd pose" and "Occluded" is decreased. This corresponds to increasing μ_0^a and μ_3^a.

As we can see in figure 7.10, both changes result in about the same performance increase. While "Improved 1" gains its increase from a better handling of a portion of negatives (that have the "Cluttered background" attribute), "Improved 2" benefits from not as much affected by two attributes that have a negative influence on positive samples.

7.4.2 Analyzing the Differences

If we look at Figure 7.10, we see that over a wide range of false positive rates, the improved detectors obtain substantially more recall than the base detector. What do

[6] We use normal distributions as the prior for the means and inverse gamma distributions as the prior for the variances.

[7] For building the FPPI score, the number of images is set to 200. Other numbers would simply shift all plots to the left or to the right.

FPPI	0.01	0.1	1	10
Occluded	18	16	21	19
Plain clothing	37	36	30	29
Bad lighting	28	27	25	35
Odd pose	11	11	12	15

(a) Improved 1 vs. Base

FPPI	0.01	0.1	1	10
Occluded	20	24	28	21
Plain clothing	39	35	31	33
Bad lighting	32	35	29	43
Odd pose	15	16	13	14

(b) Improved 2 vs. Base

Table 7.2: Result of checking the differences of thresholded sets. For each threshold (stated as a false positive rate) the set of true positives correctly identified by the improved detector, but not the base detector is formed and then checked for frequencies of the attributes. Frequencies are given as percentages. The prior probabilities for the attributes are (from top to bottom) 20%, 30%, 30% and 15% (cf. table 7.1).

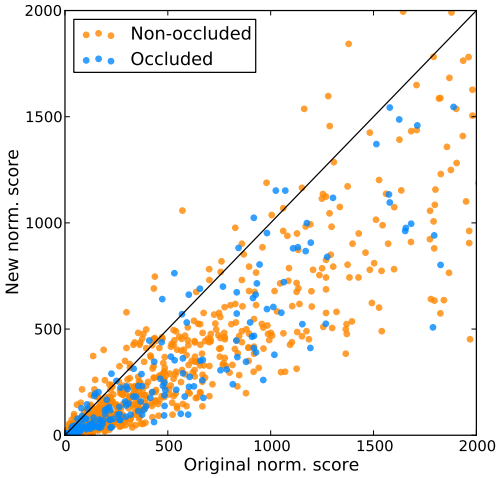
we see when we analyze the difference in recall like proposed in the introduction?

In order to perform this analysis between the base classifier and one of the improved classifiers, for a given false positive rate we first determine the set of true positives. We denote this set by TP_b for the base classifier and TP_i for the improved classifier. $TP_i \setminus TP_b$ is the set of true positives that is obtained by the improved detector but not the base detector. For 0.01, 0.1, 1 and 10 false positives per image (again with 200 as the number of images) the frequencies of the named attributes in these difference sets are shown in table 7.2. In order to reason about strengths or or weaknesses, we need to compare the values in this table with the prior probabilities from table 7.1.

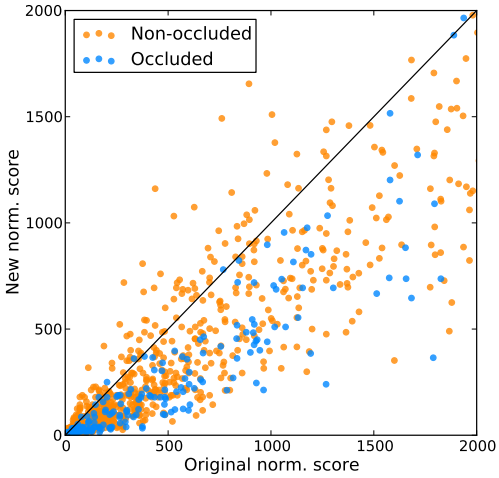
The change that we made to "Improved 1" was only related to negative samples. But when we look at table 7.2a, we see substantial deviations from prior probabilities, depending on the threshold – e.g. for 0.1 FPPI, the "Plain clothing" attribute has an about 20% stronger presence than in the total set (6 percentage points over the prior probability of 30%). At the same time, the amount of samples that have the "occluded' attribute is about 20% lower than one would expect. By using the suggested evaluation method, we are seeing patterns in the data that are not related to the real cause of change. The patterns also heavily depend on the threshold. For 1 FPPI there is no extra weight on the "Plain clothing" attribute anymore, and occluded samples are slightly more present than in the complete set.

For our "Improved 2" detector, we changed the parameters so that samples with the "Occluded" or "Odd pose" attribute get – on average – a better score. However, table 7.2a does not reflect this well at all. While the fraction of occluded samples is mostly higher than the base fraction of 20%, for 0.01 FPPI that is not the case. Also, an attribute that is unrelated to the change ("Plain Clothing") appears to have a high impact when looking at low FPPI rates.

So, we have seen that the results from this analysis are highly dependant on the threshold and that the patterns that the analysis suggest do not necessarily correlate



(a) Improved 1 vs. Base



(b) Improved 2 vs. Base

Figure 7.11: Scatter plots of old normalized score vs. new normalized score. In the left plot, the occluded and unoccluded samples are distributed evenly, while in the right plot the occluded samples have a stronger tendency to the lower right, which means they get on average a higher improvement.

with the real cause for change. One reason for this is that each threshold only results in a very narrow view on the changes, discarding all information about the samples that did not cross this threshold. If we look at low error rates, we will observe an overrepresentation of easy instances in the difference set, simply because easy samples are more likely to get high scores. An example for this is the "Plain clothing" attribute in tables 7.2a and 7.2a, which has a positive impact (so that samples having it are more likely to be easy samples). While being unrelated to the change of the detectors' properties, it is overrepresented at low error rates[8]. In a similar fashion, hard samples are likely to be overrepresented at high error rates.

While this synthetic example may seem to be contrived, the relationship that was pointed out in the last paragraph is universal: By picking a threshold, we influence the distribution of samples that are likely to cross the threshold, since the samples we will see in the difference set will be mostly samples that were just below the threshold in the base detector. A detector that is "just better" with no explicit strengths or weaknesses overall will still show patterns (overrepresentation of attributes) that can be mistaken for specifics of the detector.

To summarize, analyzing recall at a fixed error rate is not an appropriate way to make statements or to build hypotheses about a detectors' strengths relative to another one, as observed patterns can easily be artifacts of other causes, e.g. the distribution of samples near the chosen score threshold. Since the main cause of the

[8] If we extended the tables to even higher error rates, we would see an underrepresentation of this attribute.

mentioned issues is the need to select a threshold, it is natural to attempt to remove those issues by eliminating this threshold.

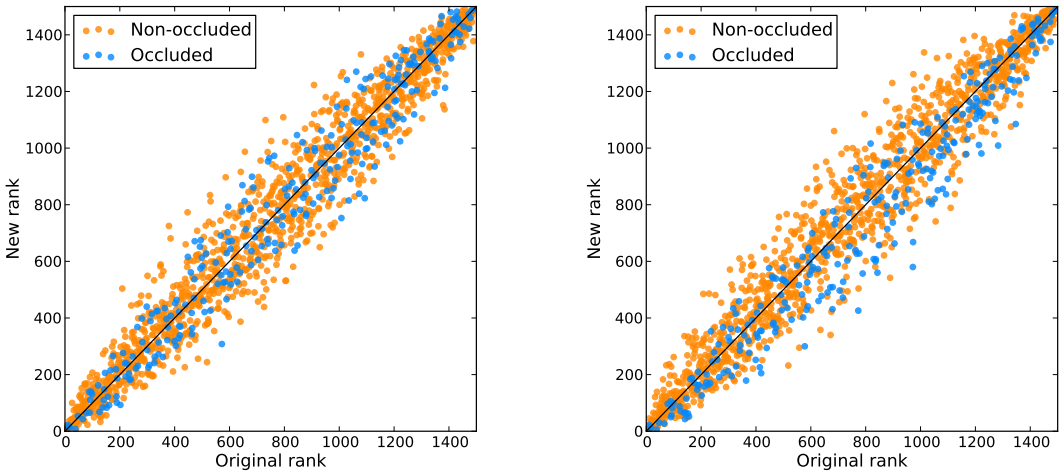
7.4.3 Instance-Level Comparison

One possible way to eliminate the arbitrary selection of a threshold would be to look at how the detection performance changes for *each* bounding box. However, simply comparing the confidence score does not work as the scores are usually not calibrated, and comparing e.g. the output of SVMs and boosting classifiers is nontrivial. However, there is one easy way to make the scores comparable for a certain test set: For each true positive we can determine how many false positives we have to accept in order to detect this instance as true positive, and for each false positive we can determine how many true positives we have to discard in order to label this instance correctly. By doing this for both detectors that we want to investigate (and assigning these numbers as scores), we can make the score comparable.[9]

Now that we made the scores comparable, we can look at differences on the instance-level instead of at the dataset-level. If we subtract the scores for the positive samples, we can give for each instance a statement of the form "For this pedestrian, we have to accept 5 more false positives in method A than in method B". Likewise, if we subtract the scores of the negative samples, we determine how much more recall has to be sacrificed to get rid of a specific false positive. One way to approach the analysis would be to see how those score differences correlate to properties of the instance. Another way to use the normalized scores is to plot the scores of the improved detectors against the scores of the old detectors in order to see how the scores changed. This was done in figure 7.11.

In both plots, the majority of data points is in the lower right half. This corresponds to the fact that the "Improved" detectors are in fact better than the "Base" detector. To see the impact of an attribute, we labeled instances with the "Occluded" attribute in blue and the other instances in orange. In figure 7.11a, where the improvement resulted from the better handling of negatives having a specific attribute, we don't see a special behavior of the occluded samples, they seem to follow the same distribution as the unoccluded samples. In contrast, in figure 7.11a, where the gain in performance is partly caused by the improved treatment of occluded samples, the occluded samples are on average drawn lower than unoccluded samples, which

[9] Strictly speaking, this does only work for the classification setting, and not for the detection setting, as detectors can miss instances completely and usually have different false positives. However, in practice this is often not much of an issue if one wants to study true positives as the overlap is usually high – for an example that we are going to show later, over 1000 pedestrians have been detected by both detectors while about 30 are only detected by one of the detectors each, mostly caused by detections on groups of people with ambiguous assignment of detections to annotations. If one wants to study false positives, it is possible to collect "hard" false positives for both detectors with a very relaxed threshold and join those sets, and use the resulting set together with the positive annotations in a classification setting.



(a) Improved 1 vs. Base

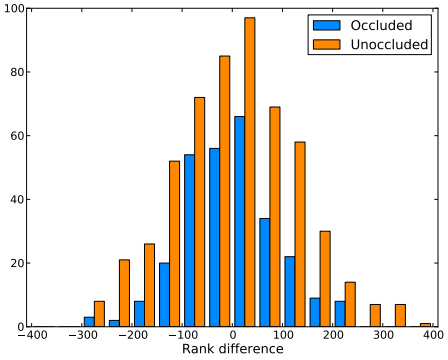
(b) Improved 2 vs. Base

Figure 7.12: Scatter plots of old rank vs. new rank. In the left image, the attribute "Occluded" did not contribute to the improvement and so samples are divided evenly between the top left and lower right halves of the plot (51% of the blue marks are in the lower right half). In the right image, the new detector does benefit from an improved handling of the "Occluded" attribute and so the majority of points is in the lower right half (68%).

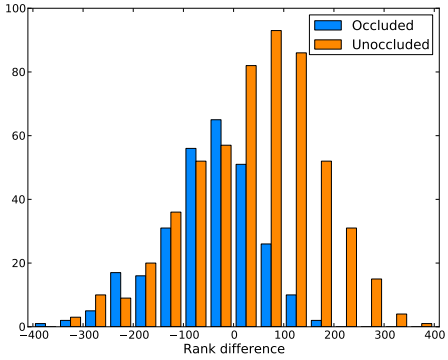
means they get on average a greater improvement. By looking at the change at the instance level, we were able to correctly identify the influence of the "Occluded" attribute (no influence in the first case, positive influence in the second), which we were not able to before.

Another way to compare detectors would be to discard the information from the negative class entirely[10] and simply look at how the detectors rank the true instances. In the same fashion as before, we assign a "score" of 1 to the true positive that the detector is most confident about, 2 to the second most confident and so on. If we then plot the new rank against the old rank, we get plots like figure 7.12. Note that this plot – in contrast to figure 7.11 – does not contain any information regarding the relative performance of the methods, so there is no way to tell which method does better. The center of mass will always be in the middle of the plot, as in the two plots in figure 7.12. However, if there are differences in which types of pedestrian a classifier considers "easy", they will show up in this plot. To demonstrate how this looks like, the samples in this plot are again colored depending on the presence of the "Occluded" label. In figure 7.12a, there is no dependence of the improvement on the label and so the occluded labels are evenly distributed on both sides of the main diagonal. In figure 7.12b, where there is a correlation between the improvement and the "Occluded" label, we can clearly see that the occluded instances tend to be in the

[10] If one wants to analyze false positives, discard the positive class.



(a) Improved 1 vs. Base



(b) Improved 2 vs. Base

Figure 7.13: Histograms of differences between old rank and new rank. In the left case, both histograms have their center around zero. In the right case, the histogram for occluded samples is shifted to the left, which means that occluded samples are easier for the detector "Improved 2" than for the detector "Base", when compared to the average pedestrian.

lower right of the plot, which means that the improved detector is more confident about the occluded samples (relative to the average sample). Another view of this data is presented in figure 7.13. There, the old rank is subtracted from the new rank, meaning that a sample that the new detector is more confident about has a negative score. The rank differences are put in a histogram. While in figure 7.13a the histograms for both occluded and unoccluded samples are centered in the middle of the plot, in figure 7.13b the histogram for the occluded samples is clearly centered in the negative range, uncovering the improved handling of occluded samples in the new detector.

The scatter plots and histograms are suited for analyzing binary (or, in general, categorical) attributes. If one wants to analyze continuous attributes, the method has to be adapted slightly. An example for a continuous attribute is shown in figure 7.14, where the size (a continuous attribute) is plotted over the rank difference that was also used in figure 7.13. The data comes from the experiment shown in figure 7.8a. The detector shown in red (using HOG, CSS, CH and HOF as the feature set) functions as our base detector. The improved detector is the detector using the same feature set with the addition of DispStat, shown in purple. As before, a negative rank difference means that the im-

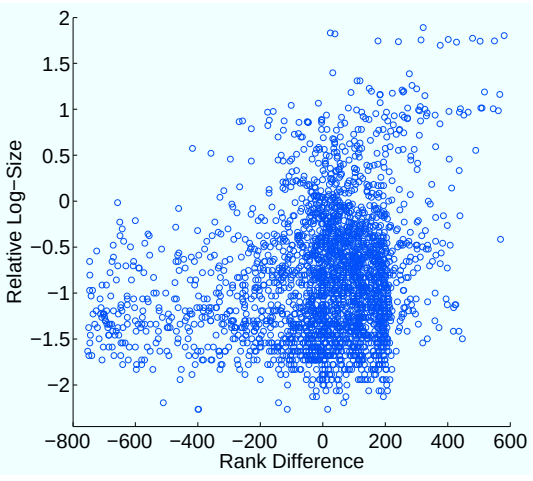


Figure 7.14: Size over rank difference

proved detector is more confident that the given instance is a positive. Interestingly, this scatter plot suggests that "big" pedestrians have a tendency to be harder for the purple detector than for the red detector, and indeed we find a positive correlation coefficient of 0.24, indicating that a connection might exist. This is counter-intuitive, as the purple detector is the one using stereo information and stereo algorithms are usually more accurate for near objects, which would mean big pedestrians. However, the existence of correlation does not prove a causal relationship, and it is plausible that something else causes the correlation. One possible explanation would be that big pedestrians are often near the image borders or clipped, and the disparity field near the image borders is flawed because of border artifacts and because it is partly extrapolated (as the second camera does not cover the entire view of our reference camera). In any case this correlation between size and relative difficulty would be interesting to investigate.

7.5 CONCLUSION

In this section we examined the influence of training sets and training procedures on detectors using high-dimensional feature vectors, and found that the unsatisfying performance of CSS and color histograms can be remedied by refined procedures, resulting in a vast overall improvement. We also argued for caution when analyzing the differences of detectors, showing that a commonly used technique can easily lead to wrong conclusions when the one using it is unaware of the issues. We proposed other methods of comparison that, while not being foolproof, will likely result in more meaningful results.

8 SUMMARY AND CONCLUSION

Contents

8.1 INTRODUCTION

This thesis advanced the state of the art in pedestrian detection in multiple ways. One focus area was building a good feature set for pedestrian detection. The choice of classifiers and how to train them was another recurring theme, as well as a view on the complete detector system. Common pitfalls in evaluation procedures that can lead to wrong conclusions have also been highlighted. The combination of these efforts led to a pedestrian detection algorithm that is consistently among the best-performing algorithms over many datasets[19]. The thesis concludes with a discussion of the contributions of this thesis and possible directions for future research in this area.

8.2 CONTRIBUTIONS

8.2.1 Features

The study of features coming from multiple sources of information has been a major focus of this thesis. In chapter 3, we have demonstrated the usefulness of

motion-based features even in settings with camera motion, where motion parallax has a significant impact on the perceived motion in the image. It was also shown that they lead to an improvement in a full-image detection setting, not only in the classification setting. In chapter 4, we further underlined the great benefit of using motion features, even if the optical flow field is degraded because of artifacts in low-quality images.

In the same chapter we introduced a new feature, color self-similarity, that utilizes color information by comparing color histograms inside the detection window, circumventing the color constancy problem that one usually has to tackle when using color. This led to a noticeable improvement particularly in the static image setting, where no motion information is available. In chapter 7 we further investigated this feature and, by using a training scheme that allowed us to increase the amount of good training data, demonstrated that it leads to a substantial improvement in all settings. Interestingly, when using the same scheme raw color histograms also improved detection rates on all tested datasets. This suggests that the unsolved color constancy problem may not be as much as a problem when enough training data is available. However, in settings where the color distribution on the test set strongly deviates from the training set, raw color will likely be harmful.

In chapter 5 a feature that utilizes a relation between scale and disparity was presented, enabling the detector to exploit scene information at the feature level. This is done in a completely data-driven way – no class-specific prior is used, contrary to the other methods we know of (which e.g. utilize a prior on human height). This feature is also complementary to depth-based HOG in the sense that utilizing the combination of the two features leads to noticeably better classification performance than using the features on their own.

8.2.2 Classifiers and Learning

We showed in chapter 3 that AdaBoost has troubles with the multi-view, multi-cue detection setting surveyed in this chapter, and that MPLBoost [4, 54] is able to overcome this issue, performing as well and sometimes even better than support vector machines. In chapter 4 we demonstrated that neglecting to employ bootstrapping correctly can lead to incorrect conclusions about relative detector performance. Chapter 5 demonstrated that combining classifiers from different families of classifiers (e.g. support vector machines and boosting) can result in increased performance, if their output is not too correlated. In this chapter we also demonstrated how to combine classifiers learned on features from different datasets, which is also used in chapter 7 to improve the performance of CSS.

8.2.3 Detector System Design

In chapter 3, we demonstrated that, in order to detect pedestrians on a small scale, upscaling the image is preferable to reducing the detector window size and presented a strategy to reduce false positives on body parts. We also improved the non-maximum suppression strategy from [10], leading to a more precise location estimation of pedestrians.

In chapter 6, we integrated the detector into the 3D scene tracking framework from [109], using multiple detectors that are restricted to partial views of the pedestrians in a mixture-of-experts model. By utilizing information from the 3D model, we could model object-object occlusions and predict which parts of the pedestrian in question are visible. This allowed us to track pedestrians which are partially occluded over extended periods of time without incurring losses in precision, as is usually the case if one uses partial detectors.

8.2.4 Evaluation Procedures

In chapter 4, we showed that the commonly used PASCAL measure suffers from an inherent bias that can lead to overestimation of object sizes. We also demonstrate that evaluating on a subset of positive detections has to be very carefully done, and that popular evaluation scripts can over- or underestimate detector performance relating to this.

In chapter 7, we demonstrate that a common way of analyzing strengths and weaknesses of detectors, which is analyzing differences in recall at a fixed precision or vice versa, can easily lead to erroneous conclusions, both missing real properties and hallucinating properties that are nonexistent. We propose other ways of analyzing the data that reduce the risk of this happening by looking at change at the instance-level, which eliminates the need to choose an arbitrary threshold.

8.3 FUTURE WORK

8.3.1 Feature Representation

The addition of new features is, together with an increase in training set size to prevent overfitting, probably the direction of steepest ascent in terms of detection performance. Also, the combination of the feature sets in this thesis with more sophisticated detectors, for example the one used by Andriluka *et al.* [2] or Felzenszwalb *et al.* [34], has not been explored, while being a promising path to a better pedestrian detector.

The integration of stronger shape cues into the feature set is an interesting avenue. Many false positives share the property that there is no pedestrian-like shape in them

and would be immediately discarded by a human because of this, yet they are often high-scoring false positives. Many features implicitly encode shape information, e.g. HOG that encodes the orientation of image gradients, however this is often done with spatial pooling of information. This is done to make the detector robust against small changes e.g. in pose, however it also prevents the classifier from picking up strong shape cues.

Since the combination of many features leads to very high feature dimensions, adding a dimensionality reduction step will likely be helpful. Schwartz *et al.* [80] suggest that a supervised dimensionality reduction step like partial least squares works better than an unsupervised one like PCA. How this compares to strong L1 regularization (in order to encourage sparsity in the used features) would be an interesting question to investigate.

8.3.2 Training sets and Training Procedures

A recurring problem during this thesis was the lack of "proper" training data. The first thing that comes to mind, which is simply adding more training data, however was not the answer. This can be seen in chapter 5, where the additional training set, which was used to learn the stereo classifiers, provided worse performance than TUD-MotionPairs. This was also the case when TUD-MotionPairs was combined with the additional training set. This was kind of counter-intuitive, because the new training set was recorded with the same setup and context (same car, same weather, same time of day, same city) as TUD-Brussels, which should make it a good training set. TUD-MotionPairs was created with very different conditions. We can do some *post hoc* reasoning about why it is not a good data set for this purpose, and try to create new datasets without the perceived weak points. However, a more systematic approach that allows a more detailed analysis would be preferable.

8.3.3 Classifiers

This thesis exclusively used binary classification between pedestrians and background. An interesting extension to this could be to use a multi-class classifier, with other classes that usually co-occur with pedestrians in images as other classes, for example cars, motorcycles or street signs. Since e.g. motorcycles are often prominent false positives, using a multi-class detector could improve detection of pedestrians. Another advantage would be that this results in more information about the objects that are present in the scene. As seen in chapter 6, information about other pedestrians in the scene can help to deal with occlusion. By also knowing about cars and other objects that may occlude pedestrians, we could cover also these types of occluders. One challenge with multi-class detectors is that they would need to cover multiple different aspect ratios of objects, which is not easily doable within the traditional sliding window approach. Structured SVMs could be one way to

overcome this problem, as the feature map it uses can be conditioned on the class label.

8.3.4 Scene Model

Aside from covering other types of occlusion or using better input detections, an interesting way to extend the scene model would be to integrate more sources of information. For example, the flow field that is computed as input for our motion feature could also serve as a cue for how objects move – in the current model, object movement is only inferred from detections and information about the motion of the camera. Utilizing a stereo setup could both improve the estimation of 3D positions and give additional cues about occlusions – both in the relative ordering of detections and the location of occlusion boundaries. Additionally a system that tracks people through occlusions over longer periods of time, possibly employing an online appearance model for each pedestrian, would be interesting.

BIBLIOGRAPHY

[1] M. Andriluka, S. Roth, and B. Schiele (2008). People-Tracking-by-Detection and People-Detection-by-Tracking, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 16

[2] M. Andriluka, S. Roth, and B. Schiele (2009). Pictorial Structures Revisited: People Detection and Articulated Pose Estimation, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 16, 23, 24, 115

[3] M. Andriluka, S. Roth, and B. Schiele (2010). Monocular 3D Pose Estimation and Tracking by Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 16

[4] B. Babenko, P. Dollár, Z. Tu, and S. Belongie (2008). Simultaneous Learning and Alignment: Multi-Instance and Multi-PoseLearning, in *ECCV Faces in Real-Life Images 2008*. 24, 28, 31, 51, 65, 114

[5] M. Bajracharya, B. Moghaddam, A. Howard, S. Brennan, and L. H. Matthies (2009). A Fast Stereo-based System for Detecting and Tracking Pedestrians from a Moving Vehicle, *The International Journal of Robotics Research*, vol. 28. 79

[6] S. Belongie, J. Malik, and J. Puzicha (2002). Shape Matching and Object Recognition Using Shape Contexts, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24(4), pp. 509–522. 13, 16

[7] C. M. Bishop (2006). *Pattern Recognition and Machine Learning*, Springer. 12

[8] W. Choi and S. Savarese (2010). Multiple Target Tracking in World Coordinate with Single, Minimally Calibrated Camera, in *Proc. of the European Conf. on Computer Vision (ECCV) 2010*. 21, 80, 88, 89, 92

[9] D. Comaniciu (2003). An Algorithm for Data-Driven Bandwidth Selection, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 25(2), pp. 281–288. 33

[10] N. Dalal (2006). *Finding People in Images and Videos*, Ph.D. thesis, Institut National Polytechnique de Grenoble. i, iii, 23, 24, 27, 29, 30, 32, 33, 37, 43, 115

[11] N. Dalal and B. Triggs (2005). Histograms of Oriented Gradients for Human Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2005*. 13, 14, 15, 16, 18, 27, 28, 29, 34, 45, 47, 51, 54, 65, 80, 81

[12] N. Dalal, B. Triggs, and C. Schmid (2006). Human Detection Using Oriented Histograms of Flow and Appearance, in *Proc. of the European Conf. on Computer Vision (ECCV) 2006*. i, iii, 18, 23, 27, 30, 43, 48, 64, 65

[13] T. Deselaers and V. Ferrari (2010). Global and Efficient Self-Similarity for Object Classification and Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 19

[14] P. Dollár, B. Babenko, S. Belongie, P. Perona, and Z. Tu (2008). Multiple Component Learning for Object Detection, in *Proc. of the European Conf. on Computer Vision (ECCV) 2008*. 16, 28

[15] P. Dollár, S. Belongie, and P. Perona (2010). The Fastest Pedestrian Detector in the West, in *Proc. of the British Machine Vision Conf. (BMVC) 2010*. 14

[16] P. Dollár, Z. Tu, P. Perona, and S. Belongie (2009). Integral channel features, in *Proc. of the British Machine Vision Conf. (BMVC) 2009*. 14, 48, 50, 51, 53, 54, 100

[17] P. Dollár, Z. Tu, H. Tao, and S. Belongie (2007). Feature Mining For Image Classification, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2007*. 13

[18] P. Dollár, C. Wojek, B. Schiele, and P. Perona (2009). Pedestrian Detection: A Benchmark, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 15, 17, 24, 25, 34, 36, 45, 46, 50, 51, 57, 58, 59

[19] P. Dollár, C. Wojek, B. Schiele, and P. Perona (2011). Pedestrian Detection: An Evaluation of the State of the Art, in *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI) 2011*. 13, 15, 17, 58, 59, 100, 113

[20] R. P. W. Duin and D. M. J. Tax (2000). Experiments with Classifier Combining Rules, in *Multiple Classifier Systems 2000*. 67

[21] M. Enzweiler, A. Eigenstetter, B. Schiele, and D. M. Gavrila (2010). Multi-Cue Pedestrian Classification With Partial Occlusion Handling, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 20, 21, 24

[22] M. Enzweiler and D. Gavrila (2008). A mixed generative-discriminative framework for pedestrian classification, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 23

[23] M. Enzweiler and D. M. Gavrila (2009). Monocular Pedestrian Detection: Survey and Experiments, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*. 11, 18, 45

[24] M. Enzweiler, P. Kanter, and D. Gavrila (2008). Monocular Pedestrian Recognition Using Motion Parallax, in *Proc. IEEE International Conf. on Intelligent Vehicles (IV) 2008*. 18, 24

[25] A. Ess, B. Leibe, K. Schindler, and L. Van Gool (2008). A Mobile Vision System for Robust Multi-Person Tracking, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 20, 33, 63, 64, 86

[26] A. Ess, B. Leibe, K. Schindler, and L. Van Gool (2009). Moving Obstacle Detection in Highly Dynamic Scenes, in *Proc. IEEE International Conf. on Robotics and Automation (ICRA) 2009*. 20

[27] A. Ess, B. Leibe, K. Schindler, and L. Van Gool (2009). Robust Multi-Person Tracking From a Mobile Platform, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 31(10), pp. 1831–1846. 20, 64, 75, 79, 80, 86, 87, 88, 92

[28] A. Ess, B. Leibe, and L. Van Gool (2007). Depth and Appearance for Mobile Scene Analysis, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2007*. 20, 33, 35, 36, 38, 39, 40, 42

[29] A. Ess, K. Schindler, B. Leibe, and L. Van Gool (2009). Improved Multi-Person Tracking with Active Occlusion Handling, in *ICRA Workshop on People Detection and Tracking 2009*. 21, 25, 79

[30] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2008). *The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results*, *http://www.pascal-network.org/challenges/VOC/voc2008/workshop/*. 57

[31] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2011). *The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results*, *http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html*. 3

[32] P. Felzenszwalb and D. Huttenlocher (2000). Efficient Matching of Pictorial Structures, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2000*. 16

[33] P. Felzenszwalb, D. McAllester, and D. Ramanan (2008). A Discriminatively Trained, Multiscale, Deformable Part Model, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 16, 28, 51, 53, 54

[34] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan (2010). Object Detection with Discriminatively Trained Part-Based Models, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32, pp. 1627–1645. 16, 24, 25, 80, 81, 87, 88, 89, 90, 91, 92, 115

[35] Y. Freund and R. Schapire (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, *Journal of Computer and System Sciences*, vol. 55, pp. 119–139. 14

[36] J. Friedman (2001). Greedy function approximation: A gradient boosting machine., *The Annals of Statistics*, vol. 29(5), pp. 1189–1232. 12

[37] J. Friedman, T. Hastie, and R. Tibshirani (2000). Additive Logistic Regression: a Statistical View of Boosting, *The Annals of Statistics*, vol. 38(2), pp. 337–374. 12, 14, 31

[38] J. Gall and V. Lempitsky (2009). Class-Specific Hough Forests for Object Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 15

[39] J. Gall, A. Yao, N. Razavi, L. van Gool, and V. Lempitsky (2011). Hough Forests for Object Detection, Tracking, and Action Recognition, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*. 15

[40] T. Gao, B. Packer, and D. Koller (2011). A Segmentation-aware Object Detection Model with Occlusion Handling, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2011*. 22, 25

[41] D. M. Gavrila (2007). A Bayesian, Exemplar-based Approach to Hierarchical Shape Matching, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*. 28

[42] D. M. Gavrila and S. Munder (2007). Multi-Cue Pedestrian Detection and Tracking from a Moving Vehicle, *International Journal of Computer Vision (IJCV)*, vol. 73, pp. 41–59. 21, 28, 63, 64, 67, 79

[43] D. M. Gavrila and V. Philomin (1999). Real-time Object Detection for "Smart" Vehicles, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 1999*. 13

[44] P. V. Gehler and S. Nowozin (2009). On Feature Combination for Multiclass Object Classification, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2009*. 38, 70

[45] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf (2010). Survey on Pedestrian Detection for Advanced Driver Assistance Systems, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32(7), pp. 1239–1258. 11

[46] J. J. Gibson (1979). *The Ecological Approach to Visual Perception*, Houghton Mifflin, Boston, MA. 27

[47] H. Hattori, A. Seki, M. Nishiyama, and T. Watanabe (2009). Stereo-Based Pedestrian Detection using Multiple Patterns, in *Proc. of the British Machine Vision Conf. (BMVC) 2009*. 20

[48] C. Huang, B. Wu, and R. Nevatia (2008). Robust object tracking by hierarchical association of detection responses, in *Proc. of the European Conf. on Computer Vision (ECCV) 2008*. 79

[49] M. Isard and J. MacCormick (2001). BraMBLe: A Bayesian Multiple-Blob Tracker, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2001*. 21, 80

[50] R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton (1991). Adaptive Mixtures of Local Experts, *Neural Computation*, vol. 3(1), pp. 79–87. 84

[51] A. Jain, T. Thormählen, H.-P. Seidel, and C. Theobalt (2010). MovieReshape: Tracking and Reshaping of Humans in Videos, vol. 29(5). 23

[52] I. N. Junejo, E. Dexter, I. Laptev, and P. Pérez (2008). Cross-View Action Recognition from Temporal Self-similarities, in *Proc. of the European Conf. on Computer Vision (ECCV) 2008*. 19

[53] R. Kaucic, A. G. Perera, G. Brooksby, J. Kaufhold, and A. Hoogs (2005). A Unified Framework for Tracking through Occlusions and Across Sensor Gaps, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2005*. 79

[54] T.-K. Kim and R. Cipolla (2008). MCBoost: Multiple Classifier Boosting for Perceptual Co-clustering of Images and Visual Features, in *Advances in Neural Information Processing Systems (NIPS) 2008*. 24, 31, 65, 114

[55] C. H. Lampert, M. B. Blaschko, and T. Hofmann (2008). Beyond Sliding Windows: Object Localization by Efficient Subwindow Search, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 17

[56] C. H. Lampert, M. B. Blaschko, and T. Hofmann (2009). Efficient Subwindow Search: A Branch and Bound Framework for Object Localization, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 31, pp. 2129–2142. 17

[57] B. Leibe and B. Schiele (2004). Scale Invariant Object Categorization Using a Scale-Adaptive Mean-Shift Search, in *Proc. of the DAGM Symposium on Pattern Recognition (DAGM) 2004*. 15

[58] B. Leibe, E. Seemann, and B. Schiele (2005). Pedestrian detection in crowded scenes, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2005*. 15

[59] Y.-Y. Lin, T.-L. Liu, and C.-S. Fuh (2004). Fast Object Detection with Occlusions, in *Proc. of the European Conf. on Computer Vision (ECCV) 2004*. 21, 25

[60] Z. Lin and L. S. Davis (2008). A Pose-Invariant Descriptor for Human Detection and Segmentation, in *Proc. of the European Conf. on Computer Vision (ECCV) 2008*. 16, 28

[61] D. G. Lowe (1999). Object Recognition from Local Scale-Invariant Features, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 1999*. 13

[62] D. G. Lowe (2004). Distinctive Image Features from Scale-Invariant Keypoints, *International Journal of Computer Vision (IJCV)*, vol. 60(2), pp. 91–110. 13

[63] S. Maji, A. Berg, and J. Malik (2008). Classification using intersection kernel SVMs is efficient, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 13, 24, 28, 31, 51, 65, 81

[64] S. Maji and J. Malik (2009). Object detection using a max-margin Hough transform, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 15

[65] J. Marin, D. Vazquez, D. Gerenimo, and A. Lopez (2010). Learning Appearance in Virtual Scenarios for Pedestrian Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 23

[66] K. Mikolajczyk, C. Schmid, and A. Zisserman (2004). Human detection based on a probabilistic assembly of robust part detectors, in *Proc. of the European Conf. on Computer Vision (ECCV) 2004*. 15

[67] A. Mohan, C. Papageorgiou, and T. Poggio (2001). Example-Based Object Detection in Images by Components, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 23(4), pp. 349–361. 14

[68] T. Ojala, M. Pietikainen, and T. Maenpaa (2002). Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24(7), pp. 971–987. 13

[69] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio (1997). Pedestrian Detection Using Wavelet Templates, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 1997*. 12

[70] P. Ott and M. Everingham (2009). Implicit Color Segmentation Features for Pedestrian and Object Detection, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2009*. 19

[71] C. Papageorgiou and T. Poggio (2000). A Trainable System for Object Detection, *International Journal of Computer Vision (IJCV)*, vol. 38(1), pp. 15–33. 12, 14, 30, 45

[72] D. Park, D. Ramanan, and C. Fowlkes (2010). Multiresolution Models for Object Detection, in *Proc. of the European Conf. on Computer Vision (ECCV) 2010*. 17, 24

[73] L. Pishchulin, C. Wojek, A. Jain, T. Thormaehlen, and B. Schiele (2011). Learning People Detection Models from Few Training Samples, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2011*. 23

[74] M. Rapus, S. Munder, G. Baratoff, and J. Denzler (2008). Pedestrian Recognition Using Combined Low-Resolution Depth and Intensity Images, in *Proc. IEEE International Conf. on Intelligent Vehicles (IV) 2008*. 20

[75] M. Rohrbach, M. Enzweiler, and D. M. Gavrila (2009). High-Level Fusion of Depth and Intensity for Pedestrian Classification, in *Proc. of the DAGM Symposium on Pattern Recognition (DAGM) 2009*. i, 20, 24, 63, 67, 71, 77

[76] P. Sabzmeydani and G. Mori (2007). Detecting Pedestrians by Learning Shapelet Features, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2007*. 14, 28

[77] P. Schnitzspan, M. Fritz, S. Roth, and B. Schiele (2009). Discriminative Structure Learning of Hierarchical Representations for Object Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 16

[78] P. Schnitzspan, S. Roth, and B. Schiele (2010). Automatic Discovery of Meaningful Object Parts with Latent CRFs, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 16, 22

[79] B. Schoelkopf and A. J. Smola (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press. 12

[80] W. Schwartz, A. Kembhavi, D. Harwood, and L. Davis (2009). Human Detection using Partial Least Squares Analysis, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2009*. 14, 48, 116

[81] E. Seemann, M. Fritz, and B. Schiele (2007). Towards Robust Pedestrian Detection in Crowded Image Sequences, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2007*. 15

[82] E. Seemann, B. Leibe, and B. Schiele (2006). Multi-Aspect Detection of Articulated Objects, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2006*. 28

[83] V. Sharma and J. Davis (2007). Integrating Appearance and Motion Cues for Simultaneous Detection and Segmentation of Pedestrians, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2007*. 18

[84] A. Shashua, Y. Gdalyahu, and G. Hayun (2004). Pedestrian Detection for Driving Assistance Systems: Single-frame Classification and System Level Performance, in *Proc. IEEE Intelligent Vehicles Symposium (IVS) 2004*. 16, 28

[85] E. Shechtman and M. Irani (2007). Matching Local Self-Similarities across Images and Videos, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2007*. 19, 24

[86] V. Shet, J. Neumann, V. Ramesh, and L. Davis (2007). Bilattice-based Logical Reasoning for Human Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2007*. 80

[87] L. Sigal and M. Black (2006). Measure Locally, Reason Globally: Occlusion-sensitive Articulated Pose Estimation, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2006*. 22

[88] Statistisches Bundesamt Deutschland (2010), *DESTATIS, http://www.destatis.de/*. 4

[89] C. Stauffer and W. E. L. Grimson (2001). Similarity templates for detection and recognition, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2001*. 19, 24

[90] A. Torralba and A. A. Efros (2011). Unbiased Look at Dataset Bias, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2011*. 23, 101

[91] D. Tran and D. Forsyth (2008). Configuration Estimates Improve Pedestrian Finding, in *Advances in Neural Information Processing Systems (NIPS) 2008*. 15

[92] O. Tuzel, F. Porikli, and P. Meer (2007). Human Detection via Classification on Riemannian Manifolds, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2007*. 14

[93] O. Tuzel, F. Porikli, and P. Meer (2008). Pedestrian Detection via Classification on Riemannian Manifolds, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 30(10), pp. 1713–1727. 14

[94] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek (2009). Evaluation of Color Descriptors for Object and Scene Recognition, in *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI) 2009*. 48

[95] M. Varma and D. Ray (2007). Learning the Discriminative Power-Invariance Trade-Off, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2007*. 28, 38

[96] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman (2009). Multiple Kernels for Object Detection, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2009*. 19

[97] A. Vedaldi and A. Zisserman (2009). Structured Output Regression for Detection with Partial Truncation, in *Advances in Neural Information Processing Systems (NIPS) 2009*. 22

[98] A. Vedaldi and A. Zisserman (2010). Efficient Additive Kernels via Explicit Feature Maps, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 13, 24

[99] P. Viola and M. Jones (2001). Fast Multi-view Face Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2001*. 18, 21

[100] P. Viola, M. Jones, and D. Snow (2003). Detecting pedestrians using patterns of motion and appearance, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2003*. i, iii, 13, 16, 18, 27, 51, 65

[101] P. Viola, M. Jones, and D. Snow (2005). Detecting pedestrians using patterns of motion and appearance, *International Journal of Computer Vision (IJCV)*, vol. 63(2), pp. 153–161. 18

[102] P. A. Viola and M. J. Jones (2004). Robust Real-Time Face Detection, *International Journal of Computer Vision (IJCV)*, vol. 57(2), pp. 137–154. 28, 31

[103] S. Walk, N. Majer, K. Schindler, and B. Schiele (2010). New features and insights for pedestrian detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2010*. 9, 46

[104] S. Walk, K. Schindler, and B. Schiele (2010). Disparity Statistics for Pedestrian Detection: Combining Appearance, Motion and Stereo, in *Proc. of the European Conf. on Computer Vision (ECCV) 2010*. 9, 102

[105] X. Wang, T. X. Han, and S. Yan (2009). An HOG-LBP Human Detector with Partial Occlusion Handling, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2009*. 13, 21, 25, 48, 99

[106] T. Watanabe, S. Ito, and K. Yokoi (2009). Co-occurrence Histograms of Oriented Gradients for Pedestrian Detection, in *PSIVT 2009*. 20, 48

[107] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof (2009). Anisotropic Huber-L1 Optical Flow, in *Proc. of the British Machine Vision Conf. (BMVC) 2009*. 48, 65

[108] J. Winn and J. Shotton (2006). The Layout Consistent Random Field for Recognizing and Segmenting Partially Occluded Objects, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2006*. 22

[109] C. Wojek, S. Roth, K. Schindler, and B. Schiele (2010). Monocular 3D Scene Modeling and Inference: Understanding Multi-Object Traffic Scenes, in *Proc. of the European Conf. on Computer Vision (ECCV) 2010*. 9, 80, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 115

[110] C. Wojek and B. Schiele (2008). A Performance Evaluation of Single and Multi-feature People Detection, in *Proc. of the DAGM Symposium on Pattern Recognition (DAGM) 2008*. 17, 24, 28, 29

[111] C. Wojek, S. Walk, S. Roth, and B. Schiele (2011). Monocular 3D Scene Understanding with Explicit Occlusion Reasoning, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2011*. 9

[112] C. Wojek, S. Walk, and B. Schiele (2009). Multi-Cue Onboard Pedestrian Detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 8, 45, 46, 48, 51, 53, 54, 55, 57, 58, 64, 65, 67, 68, 81

[113] B. Wu and R. Nevatia (2005). Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2005*. 22

[114] B. Wu and R. Nevatia (2007). Cluster Boosted Tree Classifier for Multi-View, Multi-Pose Object Detection, in *Proc. IEEE International Conf. on Computer Vision (ICCV) 2007*. 22, 28

[115] B. Wu and R. Nevatia (2007). Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet Part Detectors, *International Journal of Computer Vision (IJCV)*, vol. 75(2), pp. 247–266. 22

[116] B. Wu and R. Nevatia (2008). Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 14, 28

[117] B. Wu and R. Nevatia (2009). Detection and Segmentation of Multiple, Partially Occluded Objects by Grouping, Merging, Assigning Part Detection Responses, *International Journal of Computer Vision (IJCV)*, vol. 82(2), pp. 185–204. 22, 79

[118] B. Wu, R. Nevatia, and Y. Li (2008). Segmentation of Multiple, Partially Occluded Objects by Grouping, Merging, Assigning Part Detection Responses, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 22

[119] J. Xing, H. Ai, and S. Lao (2009). Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 22

[120] C. Zach, J.-M. Frahm, and M. Niethammer (2009). Continuous Maximal Flows and Wulff Shapes: Application to MRFs, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2009*. 75

[121] C. Zach, T. Pock, and H. Bischof (2007). A duality based approach for realtime TV-L1 optical flow, in *DAGM 2007*. 29, 30, 43, 48

[122] L. Zhang, Y. Li, and R. Nevatia (2008). Global Data Association for Multi-Object Tracking Using Network Flows, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2008*. 21, 25, 38

[123] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan (2006). Fast Human Detection Using a Cascade of Histograms of Oriented Gradients., in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2006*. 13

CURRICULUM VITAE

Name:	Stefan Walk
Date of birth:	12.11.1981
Place of birth:	Diez a.d. Lahn, Germany
Nationality:	German

Positions:

2012	ETH Zürich, Switzerland
	Research assistant at the Photogrammetry and Remote Sensing group of Prof. Dr. Konrad Schindler
2008–2011	TU Darmstadt, Germany
	PhD student at the Multimodal Interactive Systems group of Prof. Dr. Bernt Schiele

Education:

2002–2007	TU Darmstadt, Germany
	Studies of physics, graduation with the degree *Diplom-Physiker*
	Major subjects: Quantum optics and quantum information
	Diploma thesis: Photon Correlations in Spontaneous Parametric Down Conversion, supervised by Prof. Dr. Thomas Walther

PUBLICATIONS

[5] *Monocular Visual Scene Understanding: Understanding Multi-Object Traffic Scenes*
Christian Wojek, Stefan Walk, Stefan Roth, Konrad Schindler, and Bernt Schiele
In IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI),
2012, to appear

[4] *Monocular 3D Scene Understanding with Explicit Occlusion Reasoning*
Christian Wojek, Stefan Walk, Stefan Roth, and Bernt Schiele
In IEEE Conference on Computer Vision and Pattern Recognition (**CVPR**),
Colorado Springs, 2011

[3] *Disparity Statistics for Pedestrian Detection: Combining Appearance, Motion and Stereo*
Stefan Walk, Konrad Schindler, and Bernt Schiele
In European Conference on Computer Vision (ECCV),
Heraklion, 2010

[2] *New Features And Insights for Pedestrian Detection*
Stefan Walk, Nikodem Majer, Konrad Schindler, and Bernt Schiele
In IEEE Conference on Computer Vision and Pattern Recognition (**CVPR**),
San Francisco, 2010

[1] *Multi-Cue Onboard Pedestrian Detection*
Christian Wojek, Stefan Walk, and Bernt Schiele
In IEEE Conference on Computer Vision and Pattern Recognition (**CVPR**),
Miami, 2009